ESCUELA SUPERIOR DE INGENIERÍA

INGENIERÍA TÉCNICA EN INFORMÁTICA DE SISTEMAS

APPLES WARS

Sergio Marabot García

10 de Septiembre de 2012

ESCUELA SUPERIOR DE INGENIERÍA

INGENIERÍA TÉCNICA EN INFORMÁTICA DE SISTEMAS

APPLES WARS

- Departamento: Ingeniería Informática
- Directores del proyecto: Manuel Palomo Duarte y Juan Manuel Dodero Beardo
- Autor del proyecto: Sergio Marabot García

Cádiz, 10 de Septiembre de 2012

Fdo: Sergio Marabot García

Agradecimientos

Me gustaría dar las gracias a todos aquellos que me han apoyado mientras realizaba este proyecto. A todos aquellos que han tenido paciencia para escucharme hablar sobre códigos sin tener ni idea de lo que escuchaban.

También me gustaría agradecer a Manuel Palomo y a Juan Manuel Dodero por ser mis tutores y ayudarme en lo que he necesitado.

Y por último, darle un especial agradecimiento a Francisco Javier Rodríguez, no solo por su apoyo, sino por su colaboración con el diseño gráfico y sus ideas.

Licencia

Este documento ha sido liberado bajo Licencia GFDL 1.3 (GNU Free Documentation License). Se incluyen los términos de la licencia en inglés al final del documento.

Copyright © 2012 Sergio Marabot García.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Notación y formato

Al escribir este texto se ha empleado un conjunto de convenios tipográficos para facilitar la lectura y comprensión del documento:

Cuando tenemos un comando o un fragmento del código, usaremos cursivas en la notación: *comando*.

Cuando hablemos de algún programa, escribiremos el nombre en mayúsculas: Programa.

Cuando hablemos de una pantalla del juego usaremos la notación: “Pantalla”

Cuando hablemos de una opción dentro de una pantalla usaremos: “OPCION”

Índice general

1. Introducción **1**
1.1. Motivación . . . 1
1.2. Descripción del sistema actual . . . 1
1.3. Organización del documento . . . 2
2. Conceptos básicos **4**
2.1. ¿Qué es un videojuego? . . . 4
2.2. Los videojuegos como medio didáctico . . . 5
2.3. Juegos de estrategia . . . 8
3. Planificación **10**
3.1. Organización . . . 10
3.2. Diagrama de Gantt . . . 10
3.3. Costes . . . 13
4. Análisis **15**
4.1. Modelo de casos de uso . . . 15
4.1.1. Menú principal . . . 15
4.1.2. Trofeos . . . 17
4.1.3. Créditos . . . 17
4.1.4. Tutorial . . . 18
4.1.5. Opciones . . . 19
4.1.6. Juego . . . 20
4.1.7. Selección . . . 22
4.1.8. Jugando . . . 24
4.2. Modelo conceptual de datos . . . 26
4.3. Modelo de comportamiento . . . 27
4.3.1. Menú principal . . . 28
4.3.2. Trofeos . . . 34
4.3.3. Créditos . . . 35
4.3.4. Tutorial . . . 36
4.3.5. Opciones . . . 39
4.3.6. Juego . . . 45
4.3.7. Selección . . . 48
4.3.8. Jugando . . . 57
5. Diseño **62**
5.1. Gosu . . . 62
5.2. Diseño del sistema . . . 63
5.2.1. Diagrama de clases de diseño . . . 64
5.2.2. Descripción de las clases . . . 65
5.3. Diseño multimedia . . . 76
5.3.1. Personajes . . . 76
5.3.2. Mapas . . . 77
5.3.3. Música . . . 78
5.3.4. Menús . . . 78

6. Implementación **80**
6.1. Inteligencia Artificial . . . 80
6.1.1. Fácil . . . 80
6.1.2. Normal . . . 81
6.1.3. Difícil . . . 82
6.2. Colisiones . . . 84
6.3. Tablero . . . 84
6.4. Diagrama de estados de juego . . . 85
7. Pruebas **87**
7.1. Pruebas continuas . . . 87
7.2. Pruebas de errores . . . 88
8. Conclusiones **90**
8.1. Desarrollo del proyecto . . . 90
8.2. Posibles ampliaciones y mejoras . . . 91
8.3. Objetivos . . . 91

A. Guía de instalación (Linux) **93**

B. Guía del usuario **95**

C. Tecnologías implicadas **105**

D. Reglas del juego **107**

Bibliografía y referencias **109**

GNU Free Documentation License **111**
Preambles . . . 111
Terms and Conditions . . . 112
How to apply . . . 121

Índice de figuras

1.1. Logotipo de Apples Wars 1

3.1. Diagrama de Gantt (Parte 1) 11
3.2. Diagrama de Gantt (Parte 2) 11
3.3. Diagrama de Gantt (Parte 3) 12
3.4. Diagrama de Gantt (Parte 4) 12

4.1: Diagrama de caso de uso: Menú principal 15
4.2: Diagrama de caso de uso: Trofeos 17
4.3: Diagrama de caso de uso: Créditos 17
4.4: Diagrama de caso de uso: Tutorial 18
4.5: Diagrama de caso de uso: Opciones 19
4.6: Diagrama de caso de uso: Juego 21
4.7: Diagrama de caso de uso: Selección 22
4.8: Diagrama de caso de uso: Jugando 24
4.9: Diagrama de clases conceptual 27
4.10: Diagrama de secuencias del sistema: Jugar 28
4.11: Diagrama de secuencias del sistema: Trofeos 29
4.12: Diagrama de secuencias del sistema: Tutorial 30
4.13: Diagrama de secuencias del sistema: Créditos 31
4.14: Diagrama de secuencias del sistema: Opciones 32
4.15: Diagrama de secuencias del sistema: Salir 33
4.16: Diagrama de secuencias del sistema: Volver (Trofeos) 34
4.17: Diagrama de secuencias del sistema: Volver (Créditos) 35
4.18: Diagrama de secuencias del sistema: Volver (Tutorial) 36
4.19: Diagrama de secuencias del sistema: Avanzar 37
4.20: Diagrama de secuencias del sistema: Retroceder 38
4.21: Diagrama de secuencias del sistema: Volver (Opciones) 39
4.22: Diagrama de secuencias del sistema: Subir Volumen 40
4.23: Diagrama de secuencias del sistema: Bajar Volumen 41
4.24: Diagrama de secuencias del sistema: Animaciones de efectos 42
4.25: Diagrama de secuencias del sistema: Animaciones de menús 43
4.26: Diagrama de secuencias del sistema: Animaciones de movimientos 44
4.27: Diagrama de secuencias del sistema: Volver (Juego) 45
4.28: Diagrama de secuencias del sistema: Player versus Player 46
4.29: Diagrama de secuencias del sistema: Player versus CPU 47
4.30: Diagrama de secuencias del sistema: Volver (Selección) 48
4.31: Diagrama de secuencias del sistema: Mapa 49
4.32: Diagrama de secuencias del sistema: Frecuencia de magia 50
4.33: Diagrama de secuencias del sistema: Tipo de juego 51
4.34: Diagrama de secuencias del sistema: Aleatorio 52
4.35: Diagrama de secuencias del sistema: Empezar 53

4.36: Diagrama de secuencias del sistema: Jugador 1 54
4.37: Diagrama de secuencias del sistema: Jugador 2 55
4.38: Diagrama de secuencias del sistema: CPU 56
4.39: Diagrama de secuencias del sistema: Salir (Jugando) 57
4.40: Diagrama de secuencias del sistema: Activar 58
4.41: Diagrama de secuencias del sistema: Jugando 59

5.1: Diagrama de clases de diseño 64
5.2: Clase Ventana 65
5.3: Clase Tablero 67
5.4: Clase Gusano 68
5.5: Clase Animaciones 69
5.6: Clase Bloqueo 70
5.7: Clase Config_select 72
5.8: Clase Opciones 74
5.9: Clase Trofeos 75
5.10: Personajes 76
5.11: Personajes animados 77
5.12: Crisálida 77
5.13: Mariposas animadas 77
5.14: Mapas 78

6.1: IA Facil 81
6.2: IA Normal 82
6.3: IA Dificil 83
6.4: Tablero 84
6.5: Diagrama de estados de juego 85

B.1: Logo 95
B.2: Tutorial 95
B.3: Tutorial 1 96
B.4: Tutorial 2 96
B.5: Tutorial 3 97
B.6: Tutorial 4 97
B.7: Tutorial 5 98
B.8: Tutorial 6 98
B.9: Tutorial 7 99
B.10: Tutorial 8 99
B.11: Tutorial 9 100
B.12: Tutorial 10 100
B.13: Tutorial 11 101
B.14: Tutorial 12 101
B.15: Tutorial 13 102
B.16: Tutorial 14 102
B.17: Tutorial 15 103

Índice de tablas

3.1: Tabla de costes 13

Capitulo 1

Introducción

1.1. Motivación

La motivación principal que me llevó a realizar un proyecto de este tipo es querer ver más allá de un simple videojuego. Cuando eres el jugador encuentras un modo de entretenimiento, ocio, e incluso de aprendizaje, pero al hacer este proyecto he buscado ser yo el que le de a los demás un entretenimiento, un ocio o una forma de aprendizaje.

1.2. Descripción del sistema actual

Los videojuegos son parte de la sociedad hoy en día. Esta industria esta siendo llevada a todos tipos de formatos (como son los móviles), adaptados al cine e incluso llevados a los cómics.

El objetivo del PFC será crear un videojuego de estrategias en el que el jugador no sólo consiga diversión. El juego consistirá en un tablero relleno de gusanos. Cada jugador posee cierto número de gusanos, y cada uno de estos tendrá una manzana. El juego consistirá en "mandar" a los gusanos a quitarle las manzanas a los otros gusanos, para así conseguir puntos. El juego contiene 5 tipos de juegos en los que se puntúa de manera diferente. Será un juego por turnos, en el cual los jugadores mandarán a cualquier gusano (propio o del oponente) a coger una manzana de un gusano próximo. El juego se acabará cuando no haya mas movimientos posibles en el tablero, ya que los gusanos solo podrán acumular 5 manzanas. Al acumular 5 manzanas estos se transformarán en crisálidas, y al final del juego serán mariposas.

Figura 1.1: Logotipo de Apples Wars

1.3. Organización del documento

Este documento se divide en 9 capítulos:

- El primero es un capitulo introductorio. Es en el que nos encontramos.

- El segundo se describen los conceptos básicos para entender este videojuego, y las tecnologías implicadas en este.

- El tercer capitulo está dedicado a como se han organizados las tareas a realizar en el proyecto (incluyendo el diagrama de Gantt), y cuanto se ha tardado en hacerlo. También se hace una estimación del coste de este.

- El cuarto es un capitulo dedicado al análisis del juego. En el se incluye los diagramas de entidad-relación.

- El quinto está dedicado a diseño con el que se ha elaborado este videojuego.

- En el sexto hablaremos sobre la implementación del juego. En el hablaremos sobre la inteligencia artificial, sobre los estados del juego y sobre las colisiones.

- En el séptimo capitulo se explica como se han realizado las diferentes pruebas del juego durante su elaboración y posteriormente a esta.

- En el octavo hablare sobre las conclusiones a las que he llegado, cuales son las posibles mejoras o ampliaciones del juego, e incluso que he aprendido sobre los programas que he usado.

- En el noveno y último capitulo se incluye los apéndices a este documento, como puede ser parte del código del juego, la guía de instalación y la guía de usuario. También incluimos las referencias tomadas para realizar el juego y el texto de la licencia del juego.

Capitulo 2

Conceptos básicos

2.1. ¿Qué es un videojuego?

Según Wikipedia [6]: Un videojuego o juego de vídeos es un software creado para el entretenimiento en general y basado en la interacción entre una o varias personas por medio de un controlador y un aparato electrónico que ejecuta dicho videojuego.

Por lo tanto un videojuego es un software que sirve para entretener. Aún así, un videojuego se puede usar como portal para enseñar y mejorar habilidades, como pueden ser la coordinación, el razonamiento, planteamiento de estrategias, etc...

Existen diferentes tipos de juego:

- Aventura: El jugador resuelve puzles y derrota enemigos, etc....
- Educativos: Se aprende sobre una o varias temáticas.
- Disparos: El jugador debe de disparar a los enemigos, ya se con un personaje, tanque, avión , etc...
- Estrategias: El jugador debe planear como utilizar los recursos que tiene para conseguir alcanzar su objetivo.
- Lucha: El jugador maneja un personaje que pelea contra otro cuerpo a cuerpo.
- Rol: El jugador maneja personajes con ciertas características que van evolucionando con el juego.
- Musicales: Todo gira en torno a la música. Pueden ser de seguir el compás, de cantar, etc...
- Plataformas: Son juegos en los que hay que tener habilidad para pasar de unas plataformas a otras, ya sea saltando, corriendo, etc..
- Survival Horror: Son juegos de terror en los que hay que sobrevivir.
- Carreras: Son juegos en los que hay que ser el más veloz para alcanzar la meta. Suelen ser juegos de coches.
- Deportivos: giran en torno a simular un deporte.
- Simulación: Son juegos que tratan de simular una experiencia.
- Party games: Son juegos para jugar en compañía. Se suelen formar de muchos minijuegos.

2.2. Los videojuegos como medio didáctico

Al margen del potencial instructivo de los videojuegos, es decir, de lo que los jugadores pueden aprender o desarrollar mientras se divierten con el juego, existen otras investigaciones que, fundamentándose en las características de los videojuegos, estudian su posible utilización como medios didácticos en el ámbito educativo.

G. Ball (1978) publicó un artículo cuyo título es significativo: «Telegames Teach More Than You Think». En él señalaba algunos de los beneficios educativos que los videojuegos pueden proporcionar en el ámbito de los programas instructivos. Entre ellos Ball destacó los siguientes:

- Pueden ser utilizados para estimular el sentido de alerta de los jugadores y mejorar sus habilidades de pensamiento
- El formato de los videojuegos, en muchos casos, puede simular una experiencia o situación de la vida real
- Pueden incrementar el espacio de atención de aquellos jugadores que presentan dificultades en la realización de una tarea individual después de un tiempo
- Pueden ayudar en el desarrollo de habilidades para la identificación y asimilación de conceptos numéricos, reconocimiento de palabras, identificación de objetos y colores, incremento de la tasa de lectura, mejora de la comprensión…
- Algunas actividades fomentadas por los videojuegos pueden ser especialmente adecuadas para hacer frente a problemas de aprendizaje relacionados con la aceleración y el retardo
- Los jugadores pueden percibir sus propios errores y son animados a corregirlos o seleccionar otras opciones
- Los jugadores pueden transferir las conductas aprendidas a través de los videojuegos a situaciones de la vida real

Asimismo, además de destacar estas potencialidades de los videojuegos, muy aprovechables en el ámbito educativo, Ball estableció cuatro áreas para la evaluación de los videojuegos como medios didácticos: el desarrollo instructivo de los videojuegos, el desarrollo de habilidades por parte de los videojuegos, el diseño de los videojuegos y su capacidad de adaptabilidad y flexibilidad.

Años más tarde S. Long y W. Long (1984), siguiendo la teoría de Malone (1981), defendieron de nuevo los principios del aprendizaje motivador que subyacen en los videojuegos: el desafío, la fantasía y la curiosidad. Según estos autores, los videojuegos proporcionan muchos elementos aprovechables en el proceso de aprendizaje como son:

- Una implicación activa del jugador
- La opción de abandonar cuando la tarea requerida va más allá del nivel de habilidad del jugador
- Períodos cortos de actividad intensa
- Programas de tiempo flexible para el aprendizaje
- Un entorno controlable
- La posibilidad de dominio como motivador interno
- Personajes que constituyen una motivación para la mejora de destrezas y habilidades de los

jugadores tales como la coordinación óculo-manual, los tiempo de respuesta, etc

- Un feed-back continuo e inmediato, tanto visual como sonoro, que también excita, motiva y reta al jugador a un esfuerzo mayor en función del conocimiento del contenido del juego y de la comprensión de las consecuencias de las distintas estrategias de actuación
- Un proceso deductivo para la mejora de los resultados del juego ya que se omiten las causas de los errores
- Una anticipación de acontecimientos y un desarrollo de estrategias de actuación a medida que aumenta el conocimiento del videojuego

Poco después, S. Silvern (1985-86) publicó un artículo titulado «Classroom Use of Video Games» en el que, de igual modo, propugnó todo aquello que los videojuegos pueden ofrecer en términos de experiencias educativas útiles.

De acuerdo con Silvern, son las habilidades que el jugador pone en práctica las que hacen de los videojuegos una herramienta educativa valiosa. Tales habilidades pueden ordenarse de la forma siguiente:

- Ensayo y error
- Generación de un modelo o patrón
- Generación de una norma o regla
- Comprobación de la hipótesis
- Generalización
- Estimación
- Organización

Según Silvern todas ellas son necesarias para la resolución de problemas y, a menudo, no están incorporadas en el currículum escolar. Los videojuegos proporcionan una práctica en la resolución de problemas de dos formas distintas: la práctica del juego, por una parte, y el observar y compartir con otros jugadores dicha práctica para la identificación de sus soluciones, por otra.

Todo ello, en opinión de Silvern diferencia a los videojuegos de los juegos típicamente educativos. Estos últimos se limitan a la práctica de una habilidad concreta y son válidos mientras el sujeto no domina dicha habilidad, pero una vez esto ocurre, el juego educativo pierde atractivo.

Dentro del contexto de las posibilidades de utilización de los videojuegos como medio didáctico descrito, se han realizado algunas investigaciones que pretenden comprobar dicho potencial educativo.

J. Driskell y D. Dwyer (1984), interesados en las consecuencias prácticas derivadas de un sistema de formación basado en los videojuegos y en sus propiedades motivadoras, analizaron las ventajas y desventajas destacables de la formación a través de los videojuegos.

Además de destacar las ventajas motivadoras de los videojuegos presentes en los estudios de Malone (1981), revisar los procesos implicados en el aprendizaje y manifestar la falta de datos concretos sobre las propiedades de aprendizaje de los videojuegos, identificaron una serie de estrategias de posible aplicación en una formación centrada en los videojuegos:

- La contigüidad o conexión entre el estímulo y los ítems de respuesta que deben ser aprendidos. En los videojuegos la respuesta es reforzada por la velocidad y la precisión y el estímulo es contiguo en el tiempo con la respuesta deseada.

- El feedback informativo. Los videojuegos proporcionan un feedback inmediato y continuo de la respuesta correcta y de toda la actuación.
- La práctica. La velocidad y la frecuencia de respuesta en un videojuego permite la repetición de respuestas aprendidas. Esto fomenta la consistencia de la respuesta y permite un gran número de prácticas sin caer en el aburrimiento, incorporando, además, distintos niveles de dificultad.
- La adaptabilidad del contenido. El nivel de dificultad del contenido y el ritmo en un videojuego se adaptan a la actuación del usuario.
- La focalización. La complejidad de la información y los elevados ratios de respuesta de los videojuegos ayudan a focalizar la atención y promueven un nivel de implicación elevado en la actividad. Se refuerza la concentración, evitando distracciones en el aprendizaje.
- La instrucción interactiva. Los usuarios se transforman en participantes activos del proceso de aprendizaje.

Resumiendo las conclusiones de su análisis, Driskell y Dwyer propusieron tres planteamientos generales importantes en la consideración de los videojuegos como medios didácticos:

- En primer lugar, la formación centrada en los videojuegos es una formación práctica.
- En segundo lugar, las ventajas motivadoras de los videojuegos contribuyen en el aumento del interés del estudiante y la eficacia instructiva.
- Y, en último lugar, aspectos de los videojuegos como su naturaleza adaptativa e interactiva, refuerzan el proceso de aprendizaje.

K. Ricci (1994) en la introducción a su investigación, hace referencia a dos iniciativas formativas que hacen uso de los videojuegos como medio didáctico. Se trata del desarrollo de una versión de ordenador de un juego de mesa sobre la Unión Soviética diseñado para formar marines, llevada a cabo por el Naval Training Systems Center en el año 1989 y otra, del mismo centro, desarrollada en 1991 para la elaboración de bases de datos de preguntas y respuestas.

A partir de estas experiencias formativas en contextos no escolares, Ricci analizó la adquisición y retención de conocimientos básicos sobre una materia concreta presentada en formato de papel en prosa (text), en formato de pregunta-respuesta también en papel (test) y a través de un videojuego (game). Estas tres condiciones fueron seleccionadas con la intención de identificar los beneficios potenciales de los videojuegos sobre el formato de prosa tradicional y el formato de pregunta-respuesta.

A partir de una muestra de 60 estudiantes de una media de edad de 20 años distribuidos en tres grupos, uno para cada condición de aprendizaje, se hipotetizó que la condición de videojuego resultaría superior en cuanto a los resultados, puntuaciones de adquisición y retención, en comparación con los otros dos formatos.

Una vez realizado el experimento, los resultados mostraron los videojuegos como un método potencial efectivo para la adquisición y retención de conocimientos. De hecho, los sujetos asignados a la condición de videojuegos presentaron una retención superior y se centraron de forma directa en los conceptos clave. Tuvieron una participación activa en las condiciones de formación con la oportunidad de probar su conocimiento durante el proceso. Finalmente, estos sujetos puntuaron la formación recibida como más entretenida y efectiva que aquellos asignados a la otras dos condiciones.

2.3. Juegos de estrategia

Entre la diversa variedad de tipos de videojuegos que existen están los juegos de estrategias. Estos juegos requieren que el jugador ponga en práctica sus habilidades de planeamiento y pensamiento para conseguir la victoria. En la mayoría de los videojuegos de estrategia, "al jugador se le concede una vista del mundo absoluta, controlando indirectamente las unidades bajo su poder" [5].

Entre los subtipos de juegos de estrategias, nos encontramos en un TBT (turn-based tactics), un juego de tácticas por turno. En este tipo de juego el jugador tiene un periodo de análisis antes de decidir cual será su siguiente movimiento, y luego tiene que alcanzar su objetivo con las unidades que se les provee.

Las habilidades que lanzarán al jugador a la victoria son la inteligencia, habilidades técnicas y planificación y despliegue.

No podemos hablar de juego de tablero, ya que estos sólo son aquellos que tienen un tablero físico, pero nuestro juego se desarrolla sobre un tablero virtual. Al poder jugarlo contra otro jugador, el juego nos dará la oportunidad de comparar quien es el mejor estratega.

Algunos juegos de estrategias-tableros que se asemejan a Apples Wars son:

- Las damas.
- El ajedrez.

Las diferencias que existen con estos juegos son las siguientes:

- En Apples Wars el tablero está totalmente relleno inicialmente.
- Las fichas están colocadas entremezcladas, por lo que cada jugador no tendrá una zona inicial como en estos dos juegos.
- También podremos mover las fichas del oponente.
- El objetivo no es eliminar todas las fichas del enemigo como en las damas, ni eliminar la ficha más importante como en el ajedrez; El objetivo será acumular en nuestras fichas el máximo posible de manzanas (siendo el máximo 5 por cada gusano).
- Podremos usar magias que alterarán el orden natural de la partida.
- Las fichas se pueden mover a todas las direcciones (1 casilla de movimiento) siempre que haya un gusano en esa dirección, y la suma de sus manzanas no superen el máximo permitido.
- Al jugar con magias no es un juego de estrategia pura (depende del azar).

Algunas similitudes son:

- Se juega con dos jugadores.
- El tablero es de las mismas dimensiones que el tablero de ajedrez.
- Inicialmente el valor de todas las fichas es el mismo, como en las damas.
- Es un juego por turnos.
- Al jugar sin magias es un juego de estrategia pura (no depende del azar).

Capitulo 3

Planificación

3.1. Organización

A la hora de organizar el trabajo decidí hacerlo de una forma tradicional:

Lo primero a hacer en el PFC fue empezar con un planteamiento sobre lo que quería hacer. Una vez aclarado todo lo que iba a formar parte del proyecto continué con un análisis y un diseño básico. Después continué con una formación, en la cual fui aprendiendo a usar las diferentes herramientas que mas tarde tendría que usar.

Luego empecé a desarrollar la mayoría del diseño gráfica (o por lo menos todo el diseño gráfico que en aquel entonces pensé que estaría implicado). Tras finalizar el diseño gráfico empecé con la implementación del juego. Al acabar el juego comencé a documentarlo. Durante todo el proceso se realizaron las pruebas necesarias a lo que estaba desarrollando.

3.2. Diagrama de Gantt

A continuación pondré los diagramas de Gantt (en los cuales se incluye la parte de las memorias y de documentación para la presentación del PFC). En ellos podremos ver las diferentes tareas, y el tiempo que nos ha llevado realizarlas. Hay que anotar que no todos los días trabajé las mismas horas, por lo que la información sólo sirve como una aproximación a la planificación real.

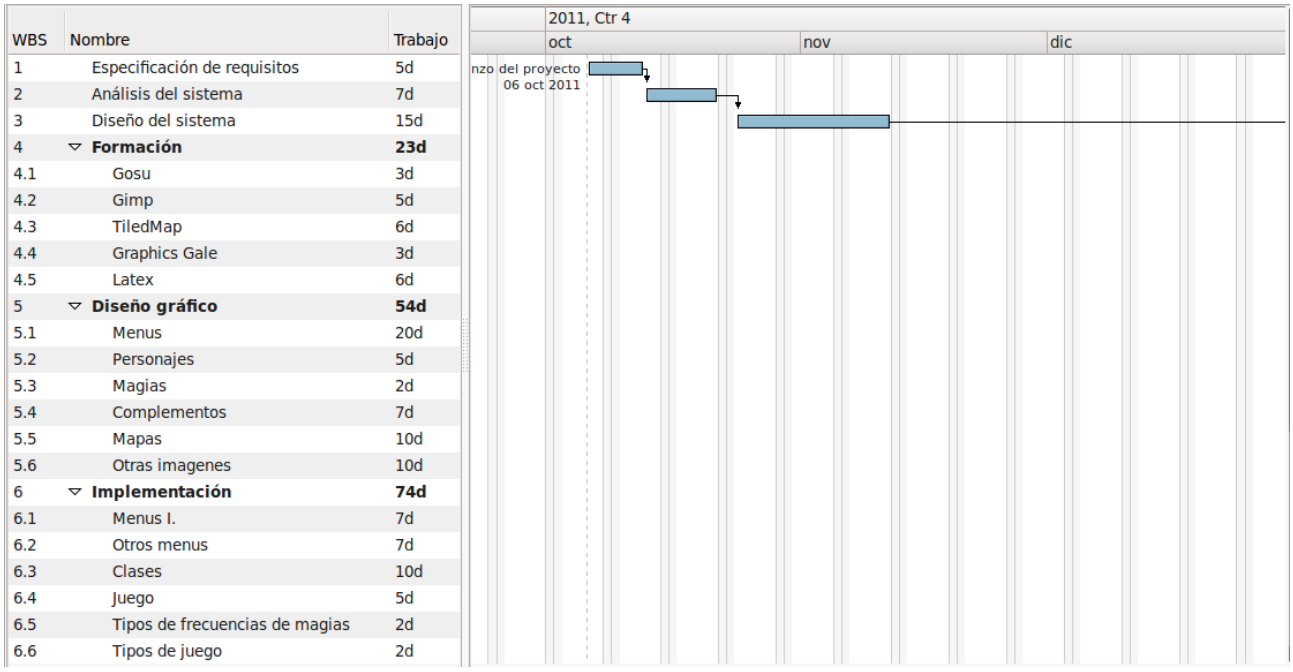

Figura 3.1: Diagrama de Gantt (Parte 1)

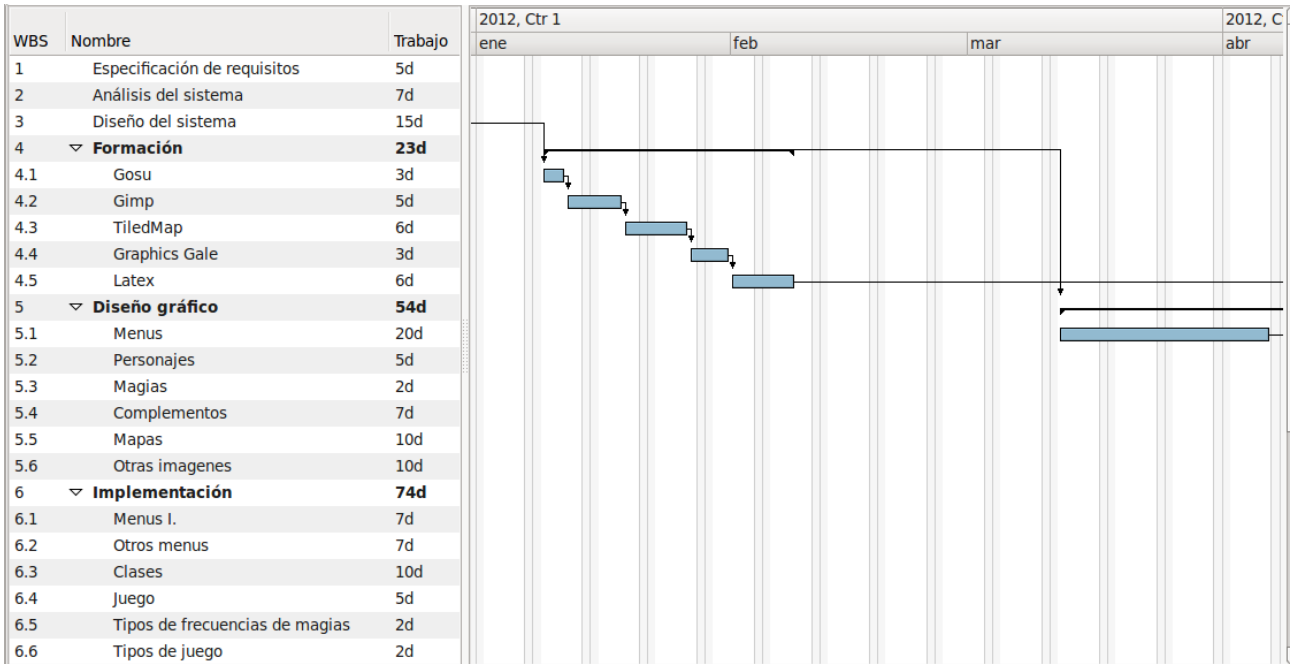

Figura 3.2: Diagrama de Gantt (Parte 2)

WBS	Nombre	Trabajo
1	Especificación de requisitos	5d
2	Análisis del sistema	7d
3	Diseño del sistema	15d
4	**Formación**	**23d**
4.1	Gosu	3d
4.2	Gimp	5d
4.3	TiledMap	6d
4.4	Graphics Gale	3d
4.5	Latex	6d
5	**Diseño gráfico**	**54d**
5.1	Menus	20d
5.2	Personajes	5d
5.3	Magias	2d
5.4	Complementos	7d
5.5	Mapas	10d
5.6	Otras imagenes	10d
6	**Implementación**	**74d**
6.1	Menus I.	7d
6.2	Otros menus	7d
6.3	Clases	10d
6.4	Juego	5d
6.5	Tipos de frecuencias de magias	2d
6.6	Tipos de juego	2d

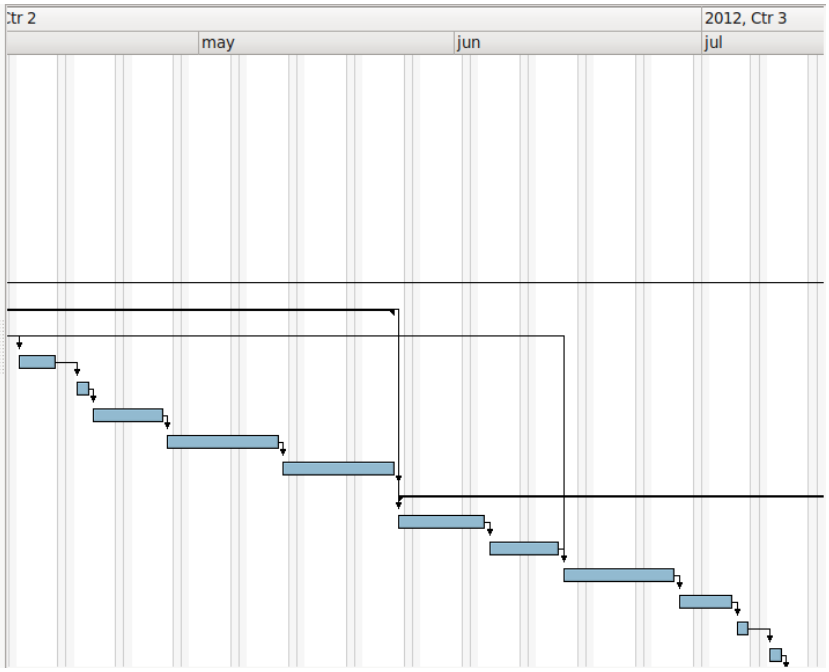

Figura 3.3: Diagrama de Gantt (Parte 3)

WBS	Nombre	Trabajo
5.3	Magias	2d
5.4	Complementos	7d
5.5	Mapas	10d
5.6	Otras imagenes	10d
6	**Implementación**	**74d**
6.1	Menus I.	7d
6.2	Otros menus	7d
6.3	Clases	10d
6.4	Juego	5d
6.5	Tipos de frecuencias de magias	2d
6.6	Tipos de juego	2d
6.7	Nombre, color y complementos	5d
6.8	Mapas I.	5d
6.9	Bloqueos	5d
6.10	Animaciones	5d
6.11	Uso magias	5d
6.12	IA	10d
6.13	Opciones	3d
6.14	Musica	3d
7	**Documentación**	**15d**
7.1	Memorias	10d
7.2	Presentación	5d
8	Pruebas	N/D

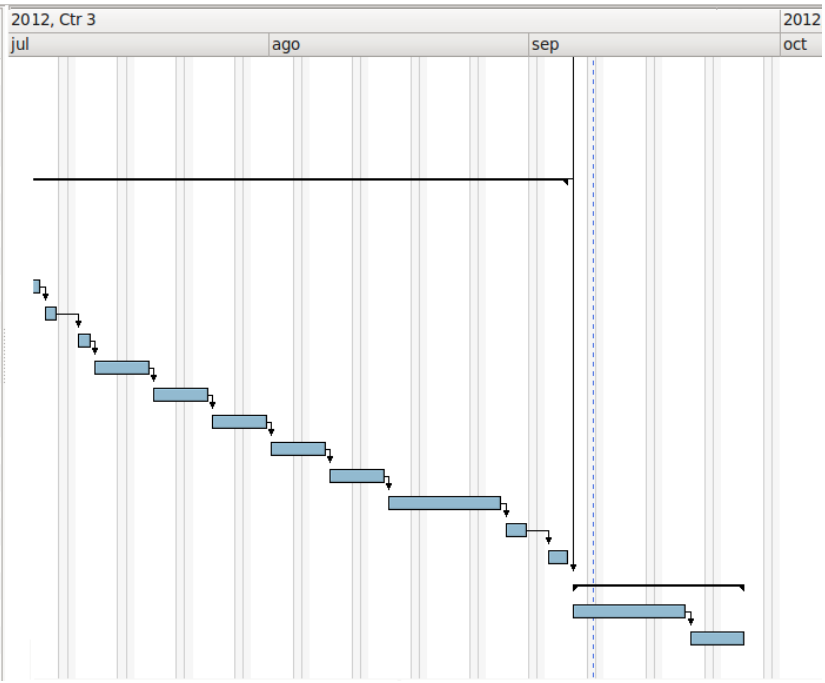

Figura 3.4: Diagrama de Gantt (Parte 4)

3.3. Costes

Respecto al coste que ha supuesto el proyecto podemos dividirlo en dos partes:

1. El coste material. La parte del diseño gráfico ha sido realizada por mí, con alguna ayuda de Francisco Javier Rodríguez, el cual lo ha realizado gratuitamente. La parte sonora del proyecto ha sido recopilado de una pagina web que distribuye música y sonidos con licencia libre y totalmente gratuita. [1]. Todo software usado es libre, por lo que no supone un gasto extra. Se podría decir que el coste material sería sólo la impresión y encuadernación de este documento, que según presupuesto de una copistería consultada sería de 12,7 euros.
2. El coste humano. Este proyecto ha sido realizado en un tiempo de 193 días, trabajando una media aproximada de 4 horas. Aún así, se ha realizado toda la programación por mí y de forma gratuita. Aún así se podría realizar una aproximación, respecto a las tablas salariales de la Universidad de Cádiz (UCA) [7].

	Días	Tiempo en meses	Salario Mensual	Total
Jornada Completa	193	9,65	1353,22	13058,57
Media Jornada	193	9,65	676,61	6529,29

Tabla 3.1: Tabla de costes

Como en este proyecto se a trabajado a media jornada el coste aproximado sería 6529,29, aunque esto sería sólo una aproximación. Añadiéndole el coste de impresión y encuadernación de este documento tendríamos un total de 6541,99 euros.

Capitulo 4

Análisis

4.1. Modelos de Caso de Uso

A continuación especificaremos el comportamiento del sistema a través de los casos de usos. Estos casos de uso están divididos dependiendo en la pantalla que te encuentres.

4.1.1. Menú Principal

El "Menú principal" será la pantalla que obtendremos al iniciar el juego. Desde aquí podremos decidir ir a otras pantallas como pueden ser: "Trofeos", "Créditos", "Tutorial", "Opciones" y "Juego".

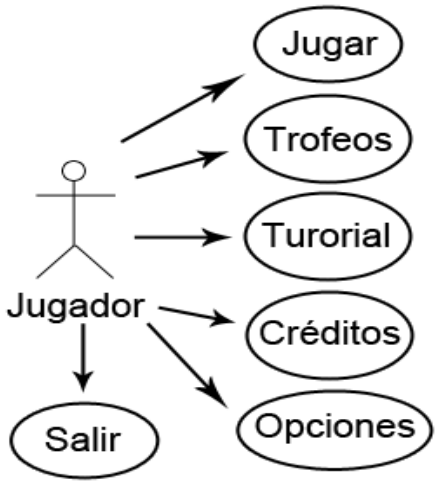

Figura 4.1: Diagrama de caso de uso: Menú principal

Caso de uso: Jugar.
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla "Menú principal".
Postcondiciones: Se muestra la pantalla "Juego".
Escenario principal:

1- El jugador selecciona la opción "JUGAR".
2- El sistema cierra la pantalla "Menú principal" e inicia la pantalla "Juego".

Escenario alternativo o de error:

* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Trofeos.
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Menú principal”.
Postcondiciones: Se muestra la pantalla “Trofeos”.
Escenario principal:
1- El jugador selecciona la opción “TROFEOS”.
2- El sistema cierra la pantalla “Menú principal” e inicia la pantalla “Trofeos”.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Tutorial.
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Menú principal”.
Postcondiciones: Se muestra la pantalla “Tutorial”.
Escenario principal:
1- El jugador selecciona la opción “TUTORIAL”.
2- El sistema cierra la pantalla “Menú principal” e inicia la pantalla “Tutorial”.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Créditos.
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Menú principal”.
Postcondiciones: Se muestra la pantalla “Créditos”.
Escenario principal:
1- El jugador selecciona la opción “CRÉDITOS”.
2- El sistema cierra la pantalla “Menú principal” e inicia la pantalla “Créditos”.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Opciones.
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Menú principal”.
Postcondiciones: Se muestra la pantalla “Opciones”.
Escenario principal:
1- El jugador selecciona la opción “OPCIONES”.
2- El sistema cierra la pantalla “Menú principal” e inicia la pantalla “Opciones”.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Salir
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Menú principal”.
Postcondiciones: Se cierra la aplicación.
Escenario principal:
1- El jugador selecciona la opción “SALIR”.
2- El sistema cierra la aplicación.
Escenario alternativo o de error:
Ninguno.

4.1.2. Trofeos

En la pantalla de “Trofeos” se nos mostrarán aquellos trofeos que hemos conseguido superando el reto correspondiente. Desde esta pantalla se puede volver a la pantalla de “Menú principal”

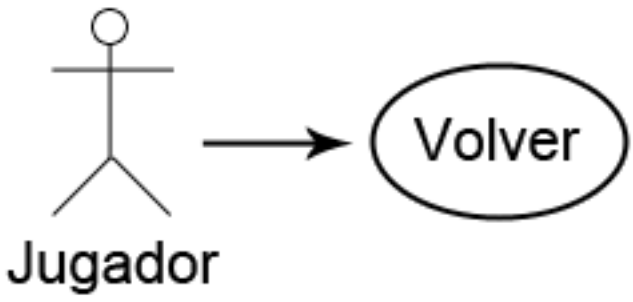

Figura 4.2: Diagrama de caso de uso: Trofeos

Caso de uso: Volver.
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Trofeos”.
Postcondiciones: Se muestra la pantalla “Menú principal”.
Escenario principal:

1- El jugador selecciona la opción “VOLVER”.
2- El sistema cierra la pantalla “Trofeos” e inicia la pantalla “Menú principal”.

Escenario alternativo o de error:

* El usuario pulsa ESC y cierra la aplicación.

4.1.3. Créditos

En la pantalla de “Créditos” se nos mostrarán los nombres de aquellos que han participado en el proyecto y lo han hecho posible. Desde esta pantalla se puede volver a la pantalla de “Menú principal”

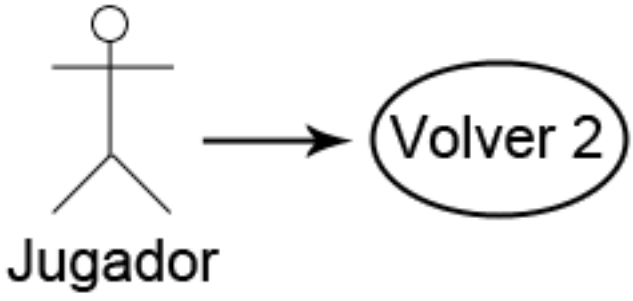

Figura 4.3: Diagrama de caso de uso: Créditos

Caso de uso: Volver 2.
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Créditos”.
Postcondiciones: Se muestra la pantalla “Menú principal”.
Escenario principal:

1- El jugador selecciona la opción “VOLVER”.
2- El sistema cierra la pantalla “Créditos” e inicia la pantalla “Menú principal”.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

4.1.4. Tutorial

En la pantalla de “Tutorial” se nos mostrará cómo usar el juego. Podremos “Avanzar” y “Retroceder” por las imágenes del tutorial. Desde esta pantalla se puede volver a la pantalla de “Menú principal”

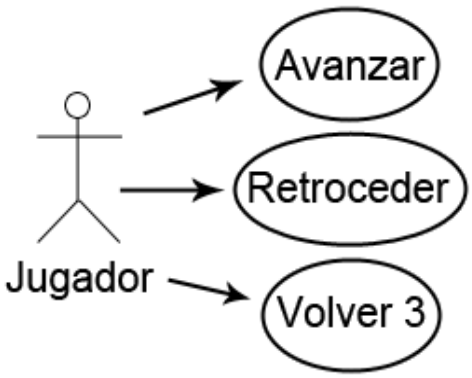

Figura 4.4: Diagrama de caso de uso: Tutorial

Caso de uso: Volver 3.
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Tutorial”.
Postcondiciones: Se muestra la pantalla “Menú principal”.
Escenario principal:
1- El jugador selecciona la opción “VOLVER”.
2- El sistema cierra la pantalla “Tutorial” e inicia la pantalla “Menú principal”.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Avanzar.
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Tutorial” y no estamos en la última imagen del tutorial.
Postcondiciones: Se muestra la siguiente imagen del tutorial.
Escenario principal:
1- El jugador selecciona la opción “AVANZAR”.
2- El sistema cambia la imagen del tutorial a la siguiente.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Retroceder.
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Tutorial” y no estamos en la primera imagen del tutorial.
Postcondiciones: Se muestra la anterior imagen del tutorial.

Escenario principal:

1- El jugador selecciona la opción “RETROCEDER”.

2- El sistema cambia la imagen del tutorial a la anterior.

Escenario alternativo o de error:

* El usuario pulsa ESC y cierra la aplicación.

4.1.5. Opciones

En la pantalla de “Opciones” se nos mostrará las distintas opciones del juego. Activar/Desactivar las distintas animaciones o subir/bajar el volumen. Desde esta pantalla se puede volver a la pantalla de “Menú principal”.

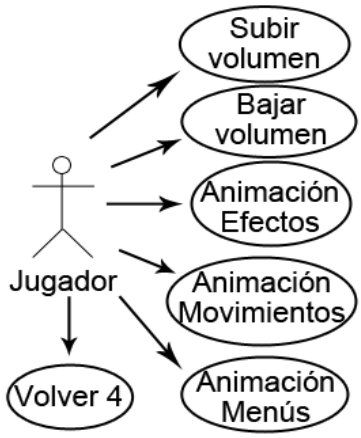

Figura 4.5: Diagrama de caso de uso: Opciones

Caso de uso: Volver 4.

Actor principal: Jugador.

Precondiciones: El juego está en la pantalla “Opciones”.

Postcondiciones: Se muestra la pantalla “Menú principal”.

Escenario principal:

1- El jugador selecciona la opción “VOLVER”.

2- El sistema cierra la pantalla “Opciones” e inicia la pantalla “Menú principal”.

Escenario alternativo o de error:

* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Subir Volumen.

Actor principal: Jugador.

Precondiciones: El juego está en la pantalla “Opciones” y el volumen no está al máximo.

Postcondiciones: Se sube el volumen

Escenario principal:

1- El jugador selecciona la opción “SUBIR VOLUMEN”.

2- El sistema sube un punto el volumen.

Escenario alternativo o de error:

* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Bajar Volumen.

Actor principal: Jugador.

Precondiciones: El juego está en la pantalla “Opciones” y el volumen no está al mínimo.

Postcondiciones: Se baja el volumen
Escenario principal:
1- El jugador selecciona la opción “BAJAR VOLUMEN”.
2- El sistema baja un punto el volumen.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Animación efectos.
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Opciones”.
Postcondiciones: Se activa/desactiva las animaciones de los efectos.
Escenario principal:
1- El jugador selecciona la opción “ANIMACIONES DE EFECTOS”.
2- El sistema desactiva las animaciones de efectos.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.
2a- El sistema activa las animaciones de efectos.

Caso de uso: Animación movimientos.
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Opciones”.
Postcondiciones: Se activa/desactiva las animaciones de los movimientos.
Escenario principal:
1- El jugador selecciona la opción “ANIMACIONES DE MOVIMIENTOS”.
2- El sistema desactiva las animaciones de movimientos.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.
2a- El sistema activa las animaciones de movimientos.

Caso de uso: Animación menús.
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Opciones”.
Postcondiciones: Se activa/desactiva las animaciones de los menús.
Escenario principal:
1- El jugador selecciona la opción “ANIMACIONES DE MENÚS”.
2- El sistema desactiva las animaciones de menús.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.
2a- El sistema activa las animaciones de menús.

4.1.6. Juego

En la pantalla de “Juego” se nos mostrará las formas de jugar al juego, ya sea contra otro jugador o contra la máquina. Desde esta pantalla se puede volver a la pantalla de “Menú principal”.

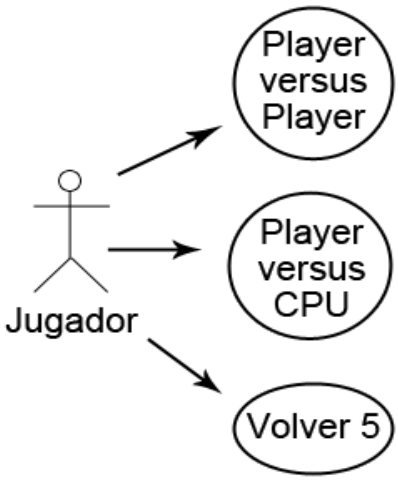

Figura 4.6: Diagrama de caso de uso: Juego

Caso de uso: Volver 5
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Juego”.
Postcondiciones: Se muestra la pantalla “Menú principal”.
Escenario principal:
1- El jugador selecciona la opción “VOLVER”.
2- El sistema cierra la pantalla “Juego” e inicia la pantalla “Menú principal”.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Player versus Player
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Juego”.
Postcondiciones: Se muestra la pantalla “Selección”.
Escenario principal:
1- El jugador selecciona la opción “PLAYER VERSUS PLAYER”.
2- El sistema cierra la pantalla “Juego” e inicia la pantalla “Selección”.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Player versus CPU
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Juego”.
Postcondiciones: Se muestra la pantalla “Selección”.
Escenario principal:
1- El jugador selecciona la opción “PLAYER VERSUS CPU”.
2- El sistema cierra la pantalla “Juego” e inicia la pantalla “Selección”.
Escenario alternativo o de error:
- El usuario pulsa ESC y cierra la aplicación.

4.1.7. Selección

En la pantalla de “Selección” se nos mostrará todas las opciones de juego. Desde esta pantalla se puede volver a la pantalla de “Juego”.

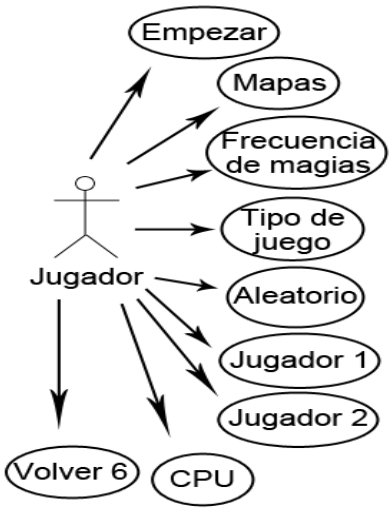

Figura 4.7: Diagrama de caso de uso: Selección

Caso de uso: Volver 6
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Selección”.
Postcondiciones: Se muestra la pantalla “Juego”.
Escenario principal:
1- El jugador selecciona la opción “VOLVER”.
2- El sistema cierra la pantalla “Selección” e inicia la pantalla “Juego”.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Mapas
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Selección”.
Postcondiciones: Se elige un nuevo mapa.
Escenario principal:
1- El jugador selecciona un mapa.
2- El sistema selecciona ese mapa como nuevo mapa.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Frecuencia de magias
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Selección”.
Postcondiciones: Se elige una nueva frecuencia de magias.
Escenario principal:
1- El jugador selecciona una frecuencia de magias.
2- El sistema selecciona esa frecuencia como nueva frecuencia de magias.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

1a- El jugador pulsa sobre la frecuencia de magias ya seleccionada.
1-El sistema quita la selección de la frecuencia de magias seleccionada.

Caso de uso: Tipo de juego
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Selección”.
Postcondiciones: Se elige un nuevo tipo de juego.
Escenario principal:
1- El jugador selecciona un tipo de juego.
2- El sistema selecciona ese tipo de juego como nuevo tipo de juego.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Aleatorio
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Selección”.
Postcondiciones: Se elige todo al azar.
Escenario principal:
1- El jugador pulsa en la opción “ALEATORIO”.
2- El sistema cambia todo al azar.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Empezar
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Selección”.
Postcondiciones: Se empieza un nuevo juego.
Escenario principal:
1- El jugador pulsa en la opción “EMPEZAR”.
2- El sistema genera un nuevo juego con las opciones seleccionadas y cambia a la pantalla “jugando”.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Jugador 1
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Selección”.
Postcondiciones: Se cambia las características del jugador 1.
Escenario principal:
1- El jugador cambia el nombre, el color y el complemento del jugador 1.
2- El sistema muestra los cambios.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.
1a- El color coincide con el del jugador 2 o la CPU.
1- El sistema no cambia el color del jugador 1.
1b- El nombre coincide con el del jugador 2.
2- El sistema asigna al jugador 1 un nombre genérico.
1c- El jugador pulsa sobre el complemento ya elegido.
1- El sistema quita la selección de ese complemento.

Caso de uso: Jugador 2
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Selección” y se ha elegido jugar en “Player versus player”.
Postcondiciones: Se cambia las características del jugador 2
Escenario principal:
1- El jugador cambia el nombre, el color y el complemento del jugador 2.
2- El sistema muestra los cambios.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.
1a- El color coincide con el del jugador 1.
1- El sistema no cambia el color del jugador 2.
1b- El nombre coincide con el del jugador 1.
2- El sistema asigna al jugador 2 un nombre genérico.
1c- El jugador pulsa sobre el complemento ya elegido.
1- El sistema quita la selección de ese complemento.

Caso de uso: CPU
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Selección” y se ha elegido jugar en “Player versus CPU”.
Postcondiciones: Se cambia las características de la CPU.
Escenario principal:
1- El jugador cambia el nombre o la dificultad de la CPU.
2- El sistema muestra los cambios.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.
1a- El color coincide con el del jugador 1.
1- El sistema no cambia el color de la CPU.

4.1.8. Jugando

En la pantalla de “Jugando” se nos mostrará el tablero de juego, y las características de la partida. Desde esta pantalla se puede volver a la pantalla de “Selección”.

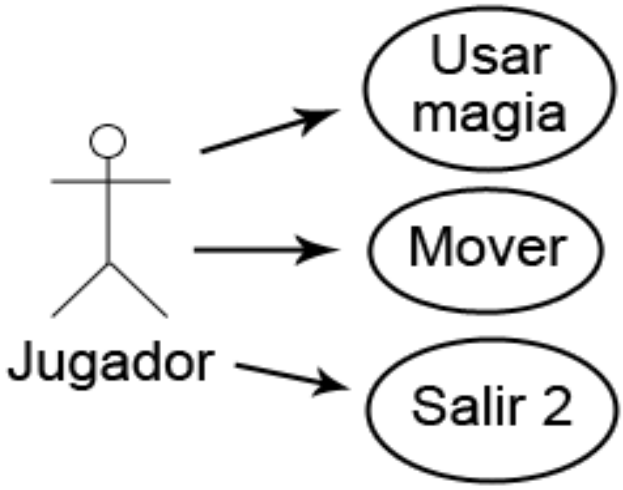

Figura 4.8: Diagrama de caso de uso: Jugando

Caso de uso: Salir 2
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Jugando”.
Postcondiciones: Se muestra la pantalla “Selección”.
Escenario principal:
1- El jugador selecciona la opción “SALIR”.
2- El sistema cierra la pantalla “Jugando” e inicia la pantalla “Selección”.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.

Caso de uso: Usar magia
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Jugando” y el jugador posee una magia.
Postcondiciones: Se realiza los cambios de la magia.
Escenario principal:
1- El jugador selecciona la opción “ACTIVAR” siendo la magia instantánea.
2- El sistema lo muestra y realiza los cambios.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.
1a- La magia no es instantánea
1- El sistema muestra las opciones disponible.
2- El jugador elige una opción.
2a- El jugador cancela la magia pulsando “DESACTIVAR”.
1-El sistema muestra los cambios.
3- El sistema lo muestra y realiza los cambios.

Caso de uso: Mover
Actor principal: Jugador.
Precondiciones: El juego está en la pantalla “Jugando” y es el turno de ese jugador.
Postcondiciones: Se realiza los cambios del movimiento del jugador.
Escenario principal:
1- El jugador elige un gusano.
2- El sistema muestra el gusano seleccionado y los posibles movimientos.
3- El jugador elige otro gusano entre los posibles.
4- El sistema lo muestra y realiza los cambios.
Escenario alternativo o de error:
* El usuario pulsa ESC y cierra la aplicación.
3a- El jugador selecciona el mismo gusano o uno que no está entre los posibles
1- El sistema quita la selección del primer gusano.

4.2. Modelo conceptual de datos

A continuación definiremos todas las clases que necesitaremos para el juego:

- **Ventana:** Clase principal del juego. Gestiona el motor gráfico, el sonido, las imágenes y la lógica del juego. Es la clase predeterminada de GOSU.

- **Tablero:** Clase que gestiona el tablero de juego. Esta clase estará compuesta por una matriz, y contendrá los gusanos con los que jugaremos.

- **Gusano:** Clase que contendrá todas las características y el comportamiento de los distintos gusanos que encontraremos durante el juego. Estos estarán colocados en el tablero anteriormente mencionado. También aparecerán en la pantalla de "Selección" para mostrarnos cuál es el resultado de las opciones escogidas para nuestros gusanos.

- **Animaciones:** Clase que se encargará de gestionar el tiempo en las animaciones y en los frames en los que se encuentran estas.

- **Bloqueo:** Clase que gestionará los distintos componentes del juego que hemos desbloqueado a medida que nos lo hemos pasado. Guardará en un fichero los avances que hemos logrado.

- **Config_select:** Clase que se encarga de memorizar la última selección realizada en la pantalla de "Selección". Guardará la información en un fichero.

- **Opciones:** Clase que se encargará de guardar en un fichero la última configuración seleccionada en la pantalla de "Opciones".

- **Trofeos:** Clase que se encargará de guardar en un fichero los distintos retos que hemos alcanzado jugando, que se nos mostrará en forma de trofeos en la pantalla "Trofeos".

A continuación añadiremos un diagrama de clases conceptual, en el que mostraremos la relación que habrá entre las distintas clases.

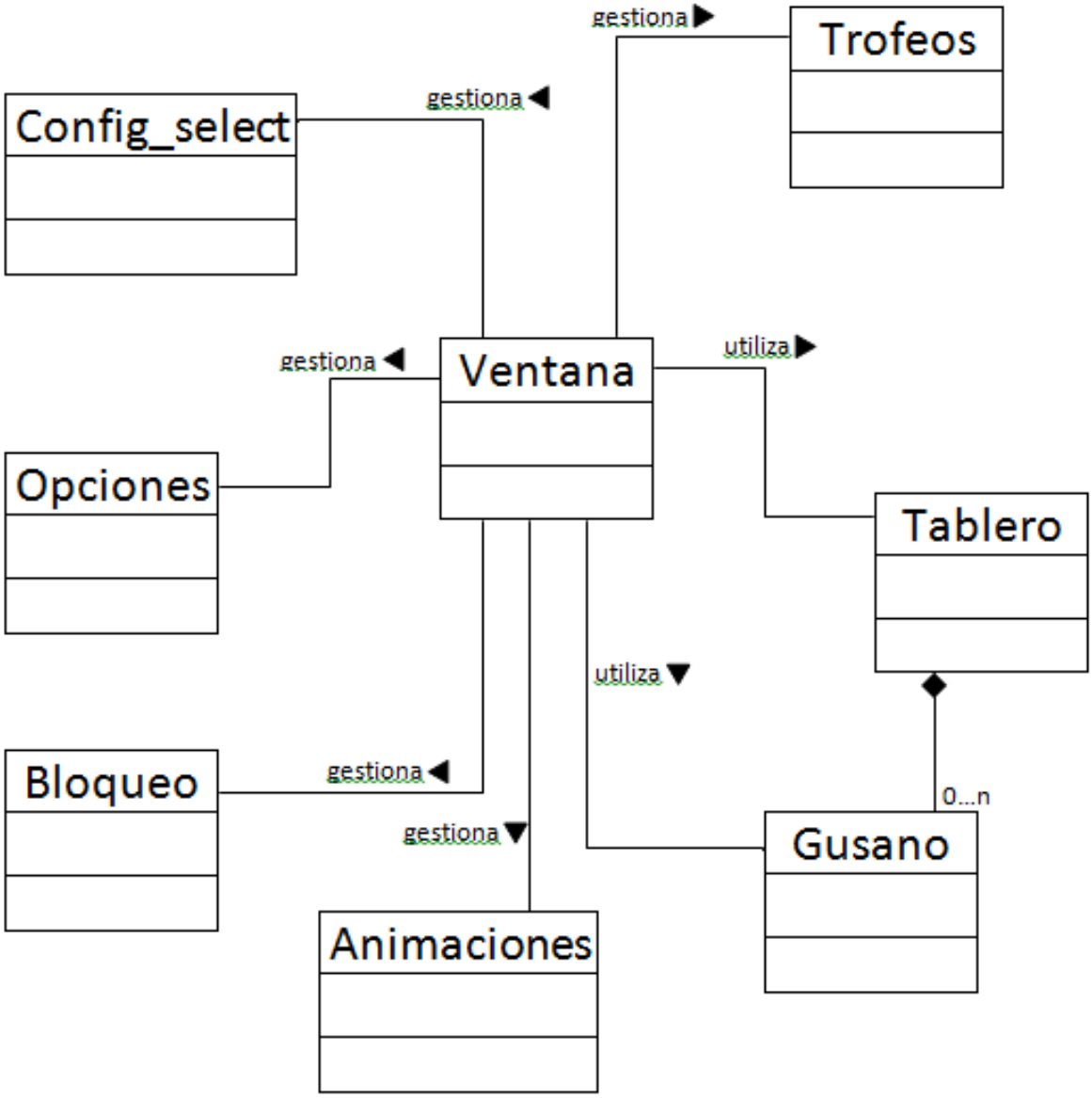

Figura 4.9: Diagrama de clases conceptual

4.3. Modelo de comportamiento

A continuación indicaremos como el sistema debe actual. Lo haremos con el modelo de comportamiento. Este se divide en dos partes:

- **Diagramas de secuencias del sistema:** muestra la secuencia de los eventos que deben ocurrir
- **Contratos de las operaciones del sistema:** describe el efecto que produce el sistema cuando realiza algún cambio.

Esta parte estará dividida en 8 partes, una por cada una de las pantallas descritas anteriormente en los modelos de casos de usos.

4.3.1. Menú principal

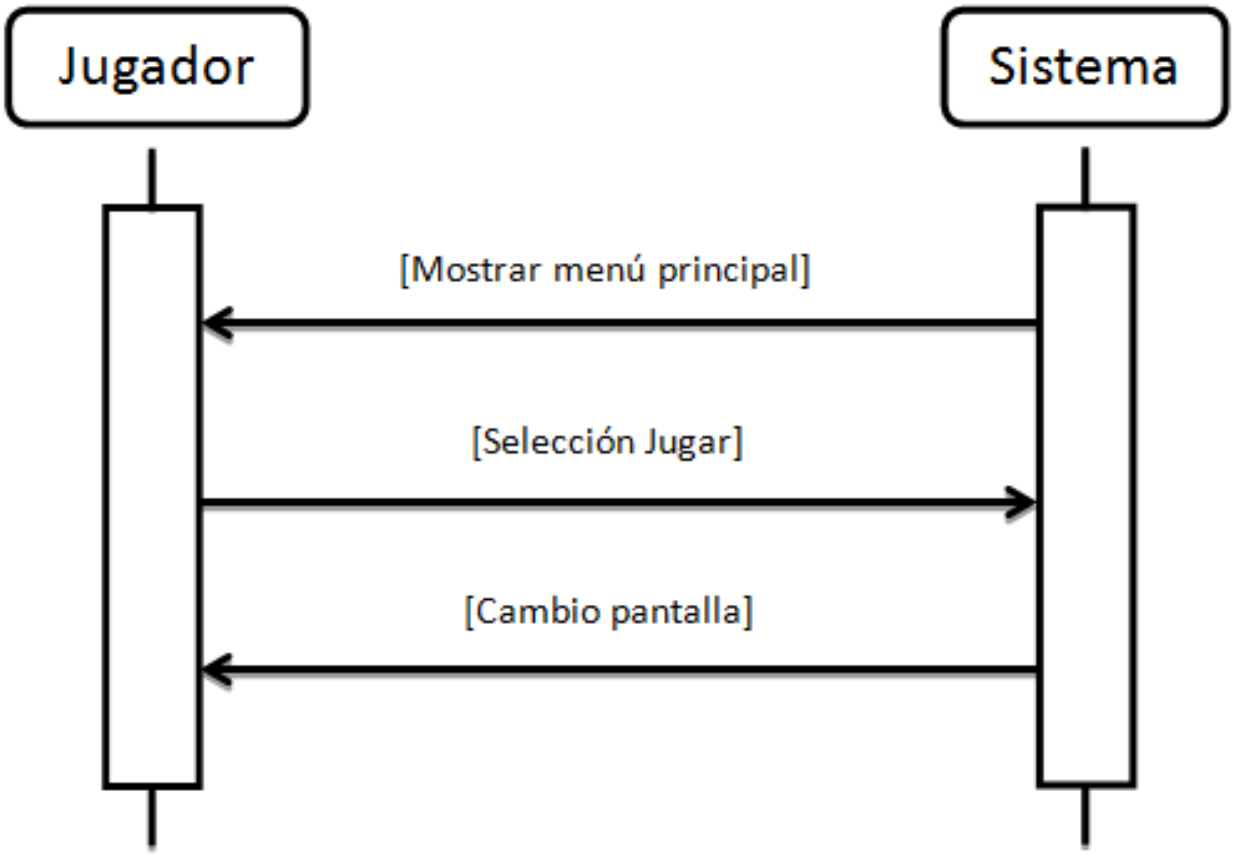

Figura 4.10: Diagrama de secuencias del sistema: Jugar

Contrato de las operaciones

Operación: Mostrar menú principal
Responsabilidad: Mostrar el menú principal con las distintas opciones a elegir.
Precondiciones: Ninguna.
Postcondiciones: Se carga todo lo necesario para mostrar el menú principal.

Operación: Selección Jugar
Responsabilidad: Seleccionar la opción "JUGAR"
Precondiciones: Se ha mostrado el menú principal.
Postcondiciones: Se ha seleccionado la opción "JUGAR"

Operación: Cambio pantalla
Responsabilidad: Se cambia de pantalla a la de "Juego".
Precondiciones: Se ha seleccionado la opción "JUGAR"
Postcondiciones: Se ha cambiado a la pantalla de "Juego".

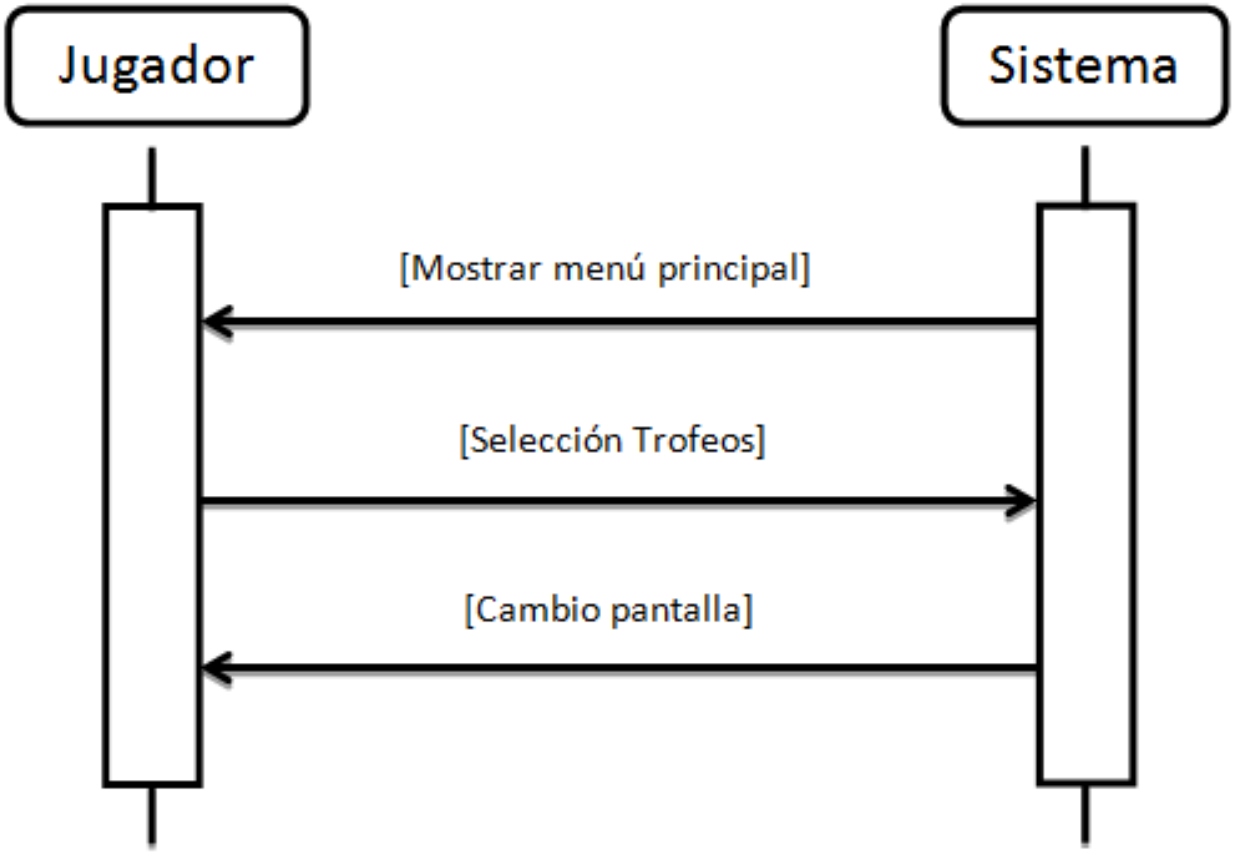

Figura 4.11: Diagrama de secuencias del sistema: Trofeos

Contrato de las operaciones

Operación: Mostrar menú principal
Responsabilidad: Mostrar el menú principal con las distintas opciones a elegir.
Precondiciones: Ninguna.
Postcondiciones: Se carga todo lo necesario para mostrar el menú principal.

Operación: Selección Trofeos
Responsabilidad: Seleccionar la opción "TROFEOS"
Precondiciones: Se ha mostrado el menú principal.
Postcondiciones: Se ha seleccionado la opción "TROFEOS"

Operación: Cambio pantalla
Responsabilidad: Se cambia de pantalla a la de "Trofeos".
Precondiciones: Se ha seleccionado la opción "TROFEOS"
Postcondiciones: Se ha cambiado a la pantalla de "Trofeos".

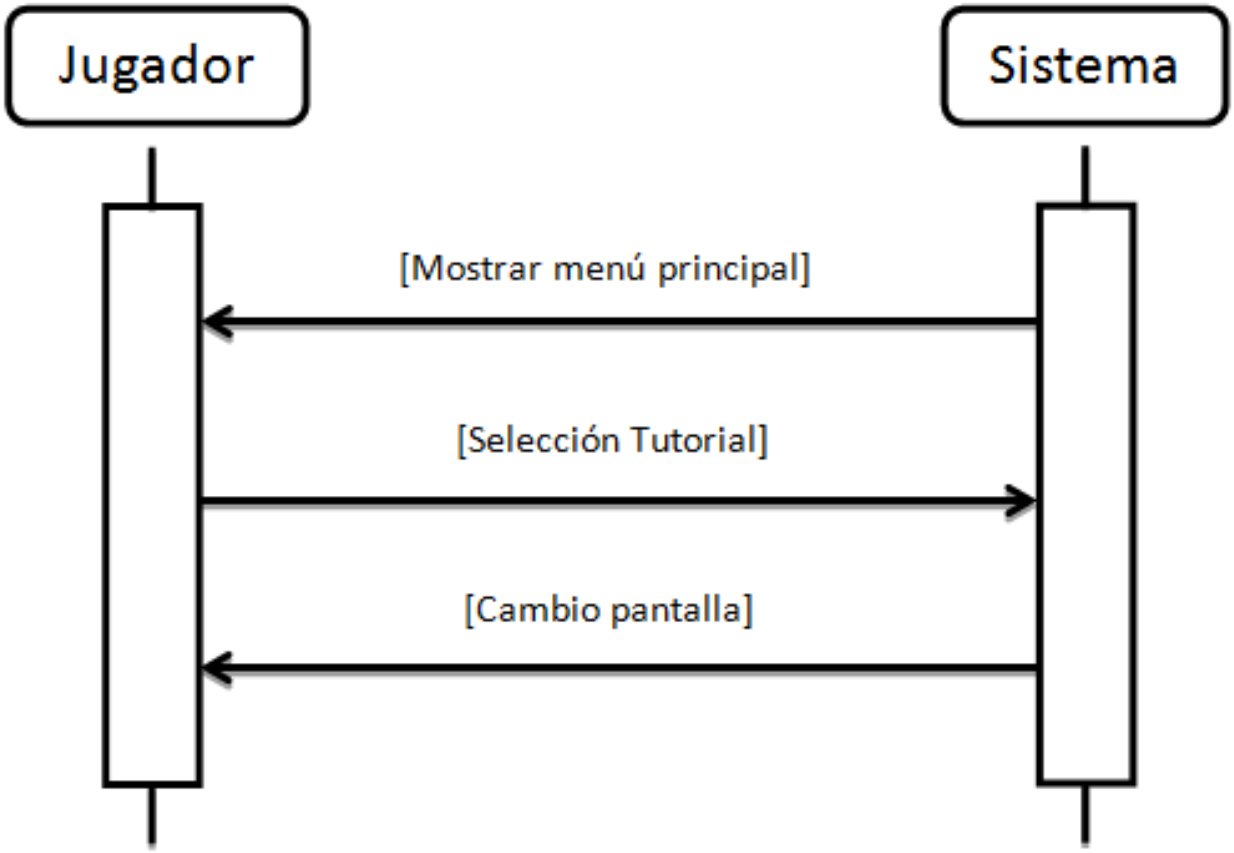

Figura 4.12: Diagrama de secuencias del sistema: Tutorial

Contrato de las operaciones

Operación: Mostrar menú principal
Responsabilidad: Mostrar el menú principal con las distintas opciones a elegir.
Precondiciones: Ninguna.
Postcondiciones: Se carga todo lo necesario para mostrar el menú principal.

Operación: Selección Tutorial
Responsabilidad: Seleccionar la opción "TUTORIAL"
Precondiciones: Se ha mostrado el menú principal.
Postcondiciones: Se ha seleccionado la opción "TUTORIAL"

Operación: Cambio pantalla
Responsabilidad: Se cambia de pantalla a la de "Tutorial".
Precondiciones: Se ha seleccionado la opción "TUTORIAL"
Postcondiciones: Se ha cambiado a la pantalla de "Tutorial".

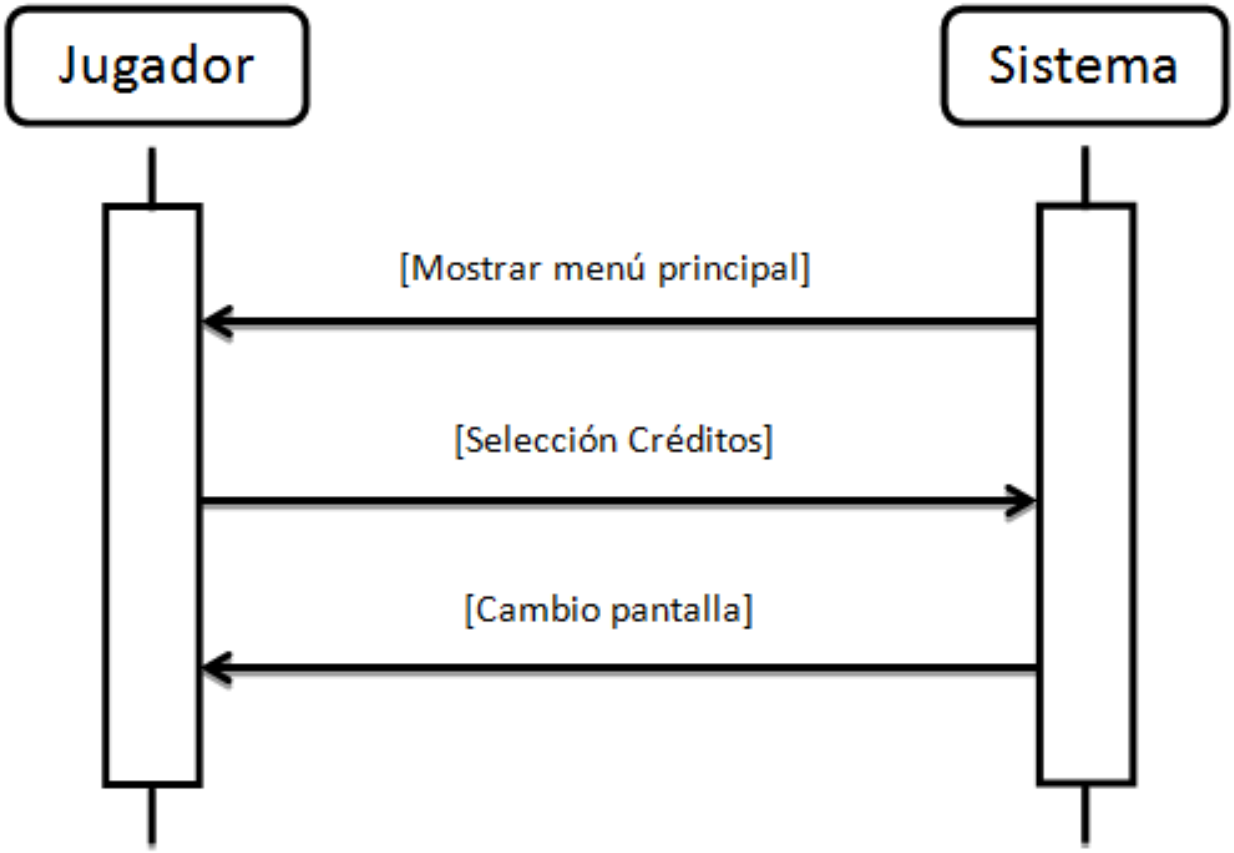

Figura 4.13: Diagrama de secuencias del sistema: Créditos

Contrato de las operaciones

Operación: Mostrar menú principal
Responsabilidad: Mostrar el menú principal con las distintas opciones a elegir.
Precondiciones: Ninguna.
Postcondiciones: Se carga todo lo necesario para mostrar el menú principal.

Operación: Selección Créditos
Responsabilidad: Seleccionar la opción "CRÉDITOS"
Precondiciones: Se ha mostrado el menú principal.
Postcondiciones: Se ha seleccionado la opción "CRÉDITOS"

Operación: Cambio pantalla
Responsabilidad: Se cambia de pantalla a la de "Créditos".
Precondiciones: Se ha seleccionado la opción "CRÉDITOS"
Postcondiciones: Se ha cambiado a la pantalla de "Créditos".

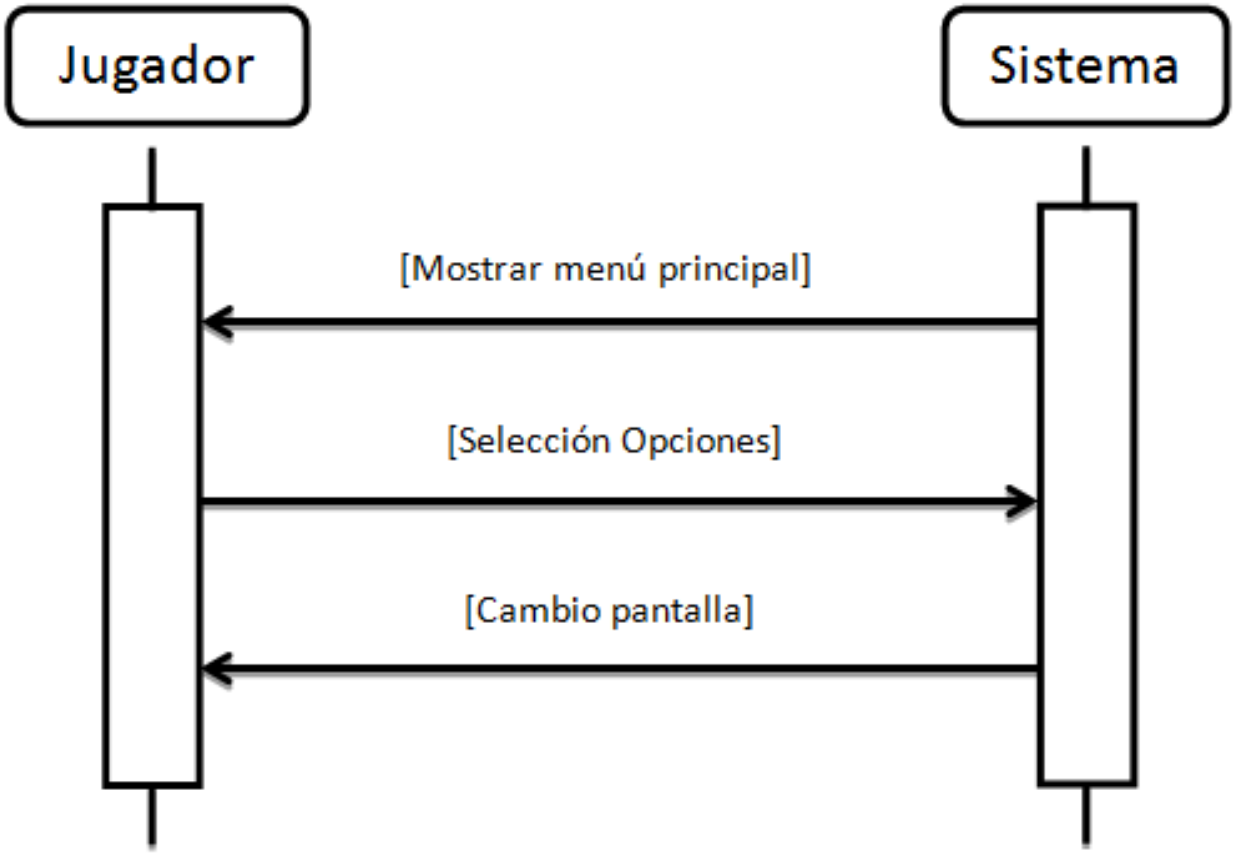

Figura 4.14: Diagrama de secuencias del sistema: Opciones

Contrato de las operaciones

Operación: Mostrar menú principal
Responsabilidad: Mostrar el menú principal con las distintas opciones a elegir.
Precondiciones: Ninguna.
Postcondiciones: Se carga todo lo necesario para mostrar el menú principal.

Operación: Selección Opciones
Responsabilidad: Seleccionar la opción “OPCIONES”
Precondiciones: Se ha mostrado el menú principal.
Postcondiciones: Se ha seleccionado la opción “OPCIONES”

Operación: Cambio pantalla
Responsabilidad: Se cambia de pantalla a la de “Opciones”.
Precondiciones: Se ha seleccionado la opción “OPCIONES”
Postcondiciones: Se ha cambiado a la pantalla de “Opciones”.

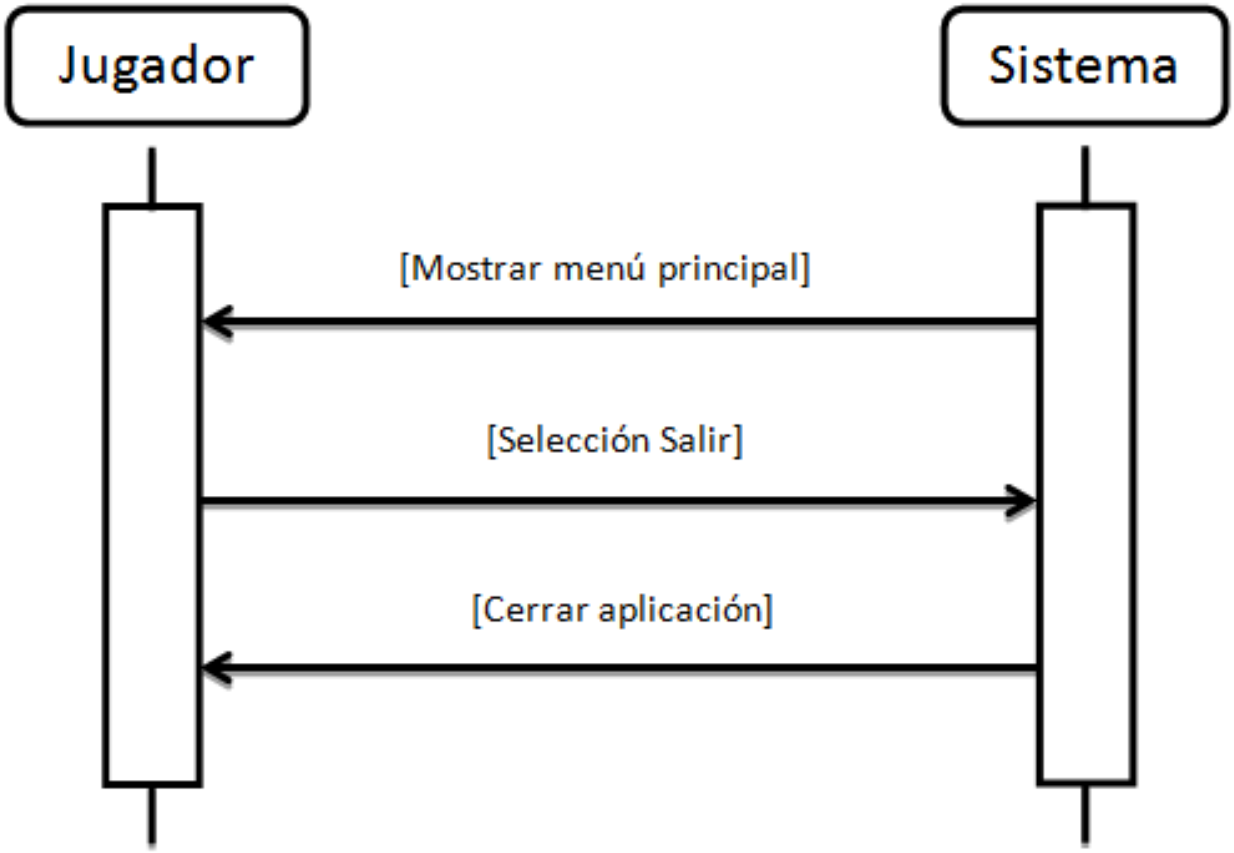

Figura 4.15: Diagrama de secuencias del sistema: Salir

Contrato de las operaciones

Operación: Mostrar menú principal
Responsabilidad: Mostrar el menú principal con las distintas opciones a elegir.
Precondiciones: Ninguna.
Postcondiciones: Se carga todo lo necesario para mostrar el menú principal.

Operación: Selección Salir
Responsabilidad: Seleccionar la opción “SALIR”
Precondiciones: Se ha mostrado el menú principal.
Postcondiciones: Se ha seleccionado la opción “SALIR”

Operación: Cerrar aplicación
Responsabilidad: Se cierra la aplicación liberando la memoria usada.
Precondiciones: Se ha seleccionado la opción “SALIR”
Postcondiciones: Se libera la memoria usada y se cierra la aplicación.

4.3.2. Trofeos

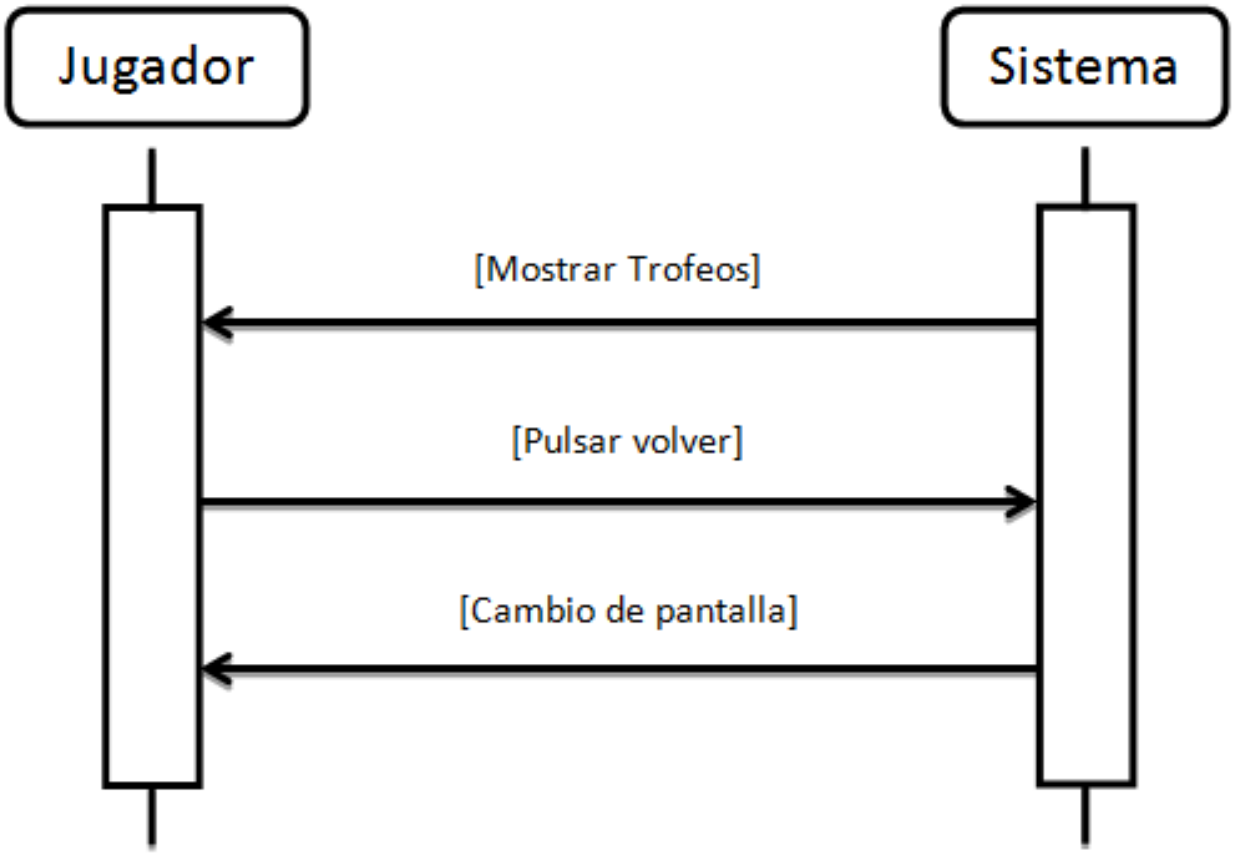

Figura 4.16: Diagrama de secuencias del sistema: Volver (Trofeos)

Contrato de las operaciones

Operación: Mostrar trofeos
Responsabilidad: Mostrar los trofeos conseguidos.
Precondiciones: Se ha entrado en la pantalla de “Trofeos”.
Postcondiciones: Se carga todo lo necesario para mostrar los trofeos.

Operación: Pulsar volver
Responsabilidad: Seleccionar la opción “VOLVER”
Precondiciones: Se ha mostrado la pantalla de “Trofeos”.
Postcondiciones: Se ha seleccionado la opción “VOLVER”

Operación: Cambio de pantalla
Responsabilidad: Se cambia de pantalla a la de “Menú principal”.
Precondiciones: Se ha seleccionado la opción “VOLVER”
Postcondiciones: Se ha cambiado a la pantalla de “Menú principal”.

4.3.3. Créditos

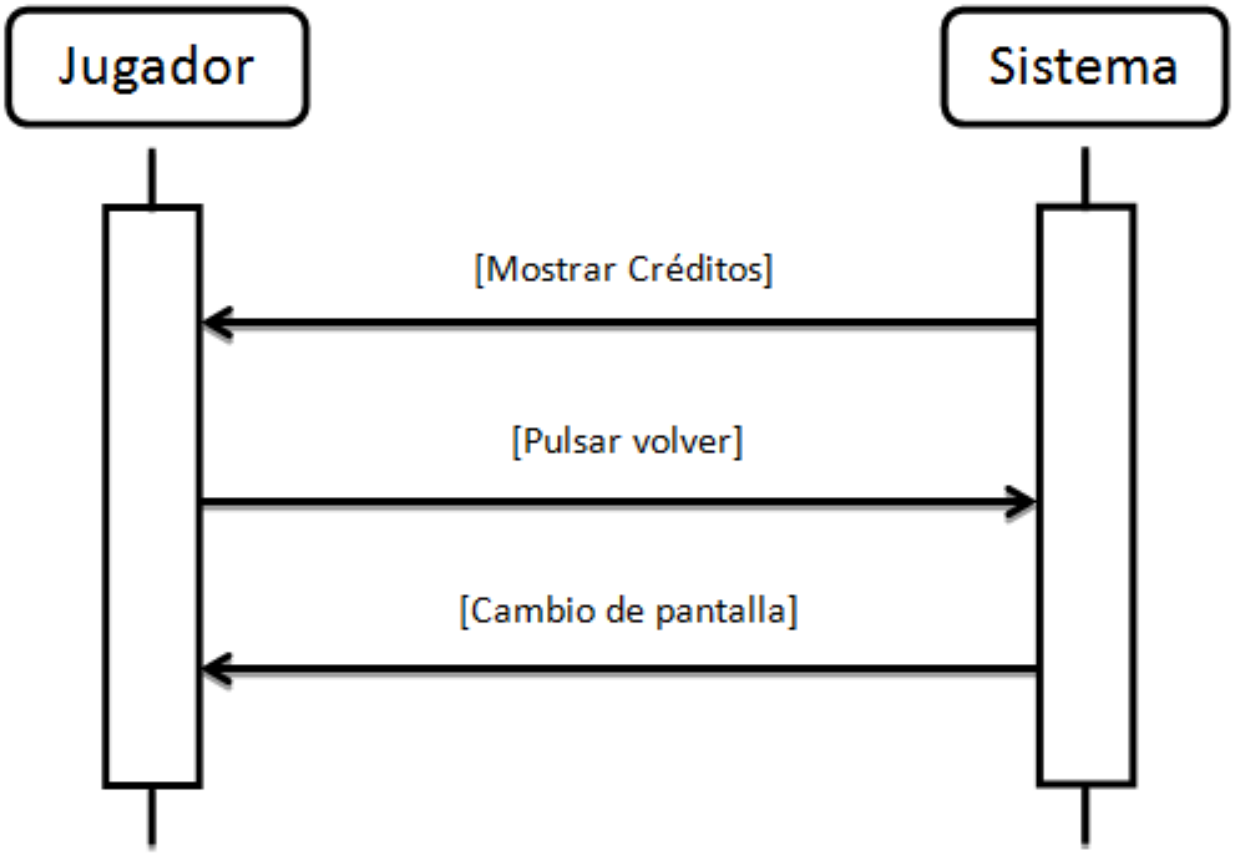

Figura 4.17: Diagrama de secuencias del sistema: Volver (Créditos)

Contrato de las operaciones

Operación: Mostrar Créditos
Responsabilidad: Mostrar los créditos.
Precondiciones: Se ha entrado en la pantalla de “Créditos”.
Postcondiciones: Se carga todo lo necesario para mostrar los créditos.

Operación: Pulsar volver
Responsabilidad: Seleccionar la opción “VOLVER”
Precondiciones: Se ha mostrado la pantalla de “Créditos”.
Postcondiciones: Se ha seleccionado la opción “VOLVER”

Operación: Cambio de pantalla
Responsabilidad: Se cambia de pantalla a la de “Menú principal”.
Precondiciones: Se ha seleccionado la opción “VOLVER”
Postcondiciones: Se ha cambiado a la pantalla de “Menú principal”.

4.3.4. Tutorial

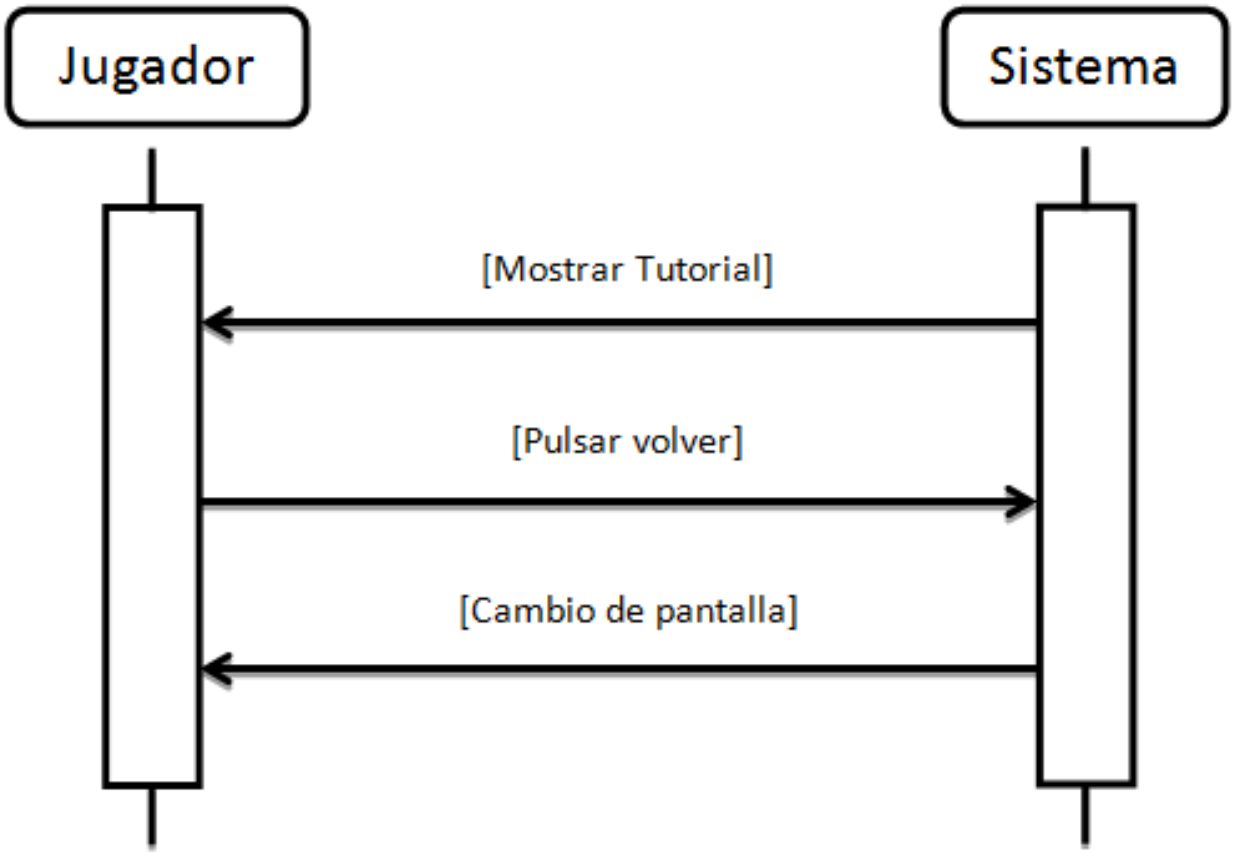

Figura 4.18: Diagrama de secuencias del sistema: Volver (Tutorial)

Contrato de las operaciones

Operación: Mostrar Tutorial
Responsabilidad: Mostrar el tutorial.
Precondiciones: Se ha entrado en la pantalla de “Tutorial”.
Postcondiciones: Se carga todo lo necesario para mostrar el tutorial.

Operación: Pulsar volver
Responsabilidad: Seleccionar la opción “VOLVER”
Precondiciones: Se ha mostrado la pantalla de “Tutorial”.
Postcondiciones: Se ha seleccionado la opción “VOLVER”

Operación: Cambio de pantalla
Responsabilidad: Se cambia de pantalla a la de “Menú principal”.
Precondiciones: Se ha seleccionado la opción “VOLVER”
Postcondiciones: Se ha cambiado a la pantalla de “Menú principal”.

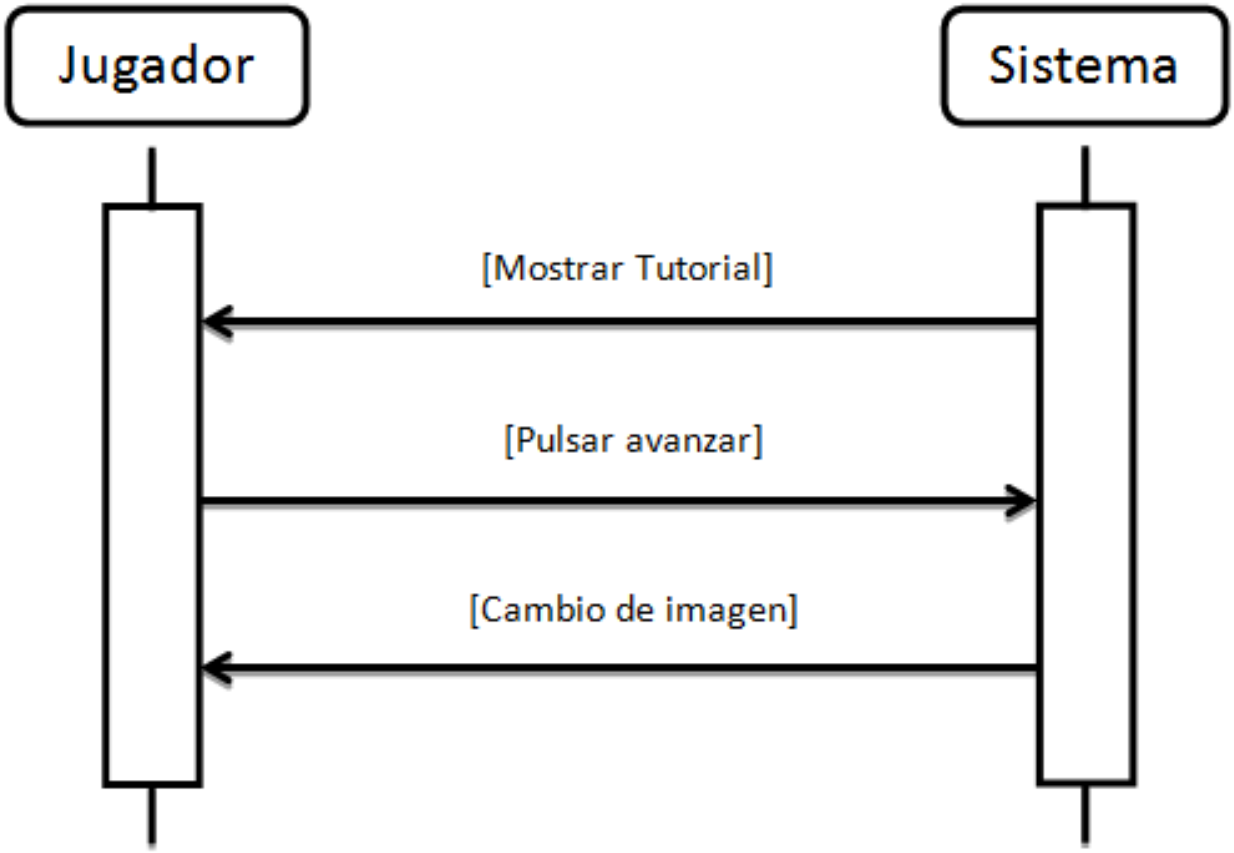

Figura 4.19: Diagrama de secuencias del sistema: Avanzar

Contrato de las operaciones

Operación: Mostrar Tutorial
Responsabilidad: Mostrar el tutorial.
Precondiciones: Se ha entrado en la pantalla de “Tutorial”.
Postcondiciones: Se carga todo lo necesario para mostrar el tutorial.

Operación: Pulsar avanzar
Responsabilidad: Seleccionar la opción “AVANZAR”
Precondiciones: Se ha mostrado la pantalla de “Tutorial”.
Postcondiciones: Se ha seleccionado la opción “AVANZAR”

Operación: Cambio de imagen
Responsabilidad: Se cambia a la siguiente imagen.
Precondiciones: Se ha seleccionado la opción “AVANZAR” y no estamos en la última imagen.
Postcondiciones: Se ha cambiado a la siguiente imagen.

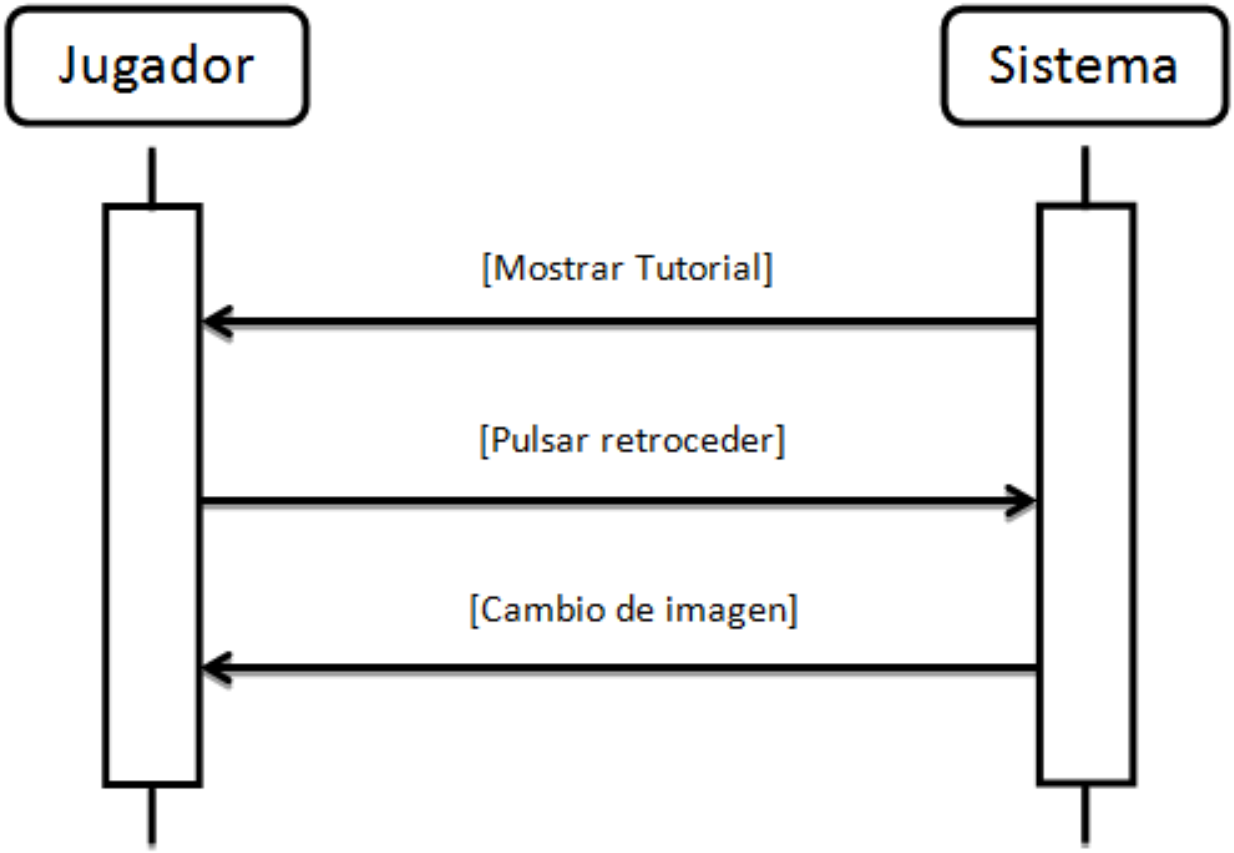

Figura 4.20: Diagrama de secuencias del sistema: Retroceder

Contrato de las operaciones

Operación: Mostrar Tutorial
Responsabilidad: Mostrar el tutorial.
Precondiciones: Se ha entrado en la pantalla de "Tutorial".
Postcondiciones: Se carga todo lo necesario para mostrar el tutorial.

Operación: Pulsar avanzar
Responsabilidad: Seleccionar la opción "RETROCEDER"
Precondiciones: Se ha mostrado la pantalla de "Tutorial".
Postcondiciones: Se ha seleccionado la opción "RETROCEDER"

Operación: Cambio de imagen
Responsabilidad: Se cambia a la anterior imagen.
Precondiciones: Se ha seleccionado la opción "RETROCEDER" y no estamos en la primera imagen.
Postcondiciones: Se ha cambiado a la anterior imagen.

4.3.5. Opciones

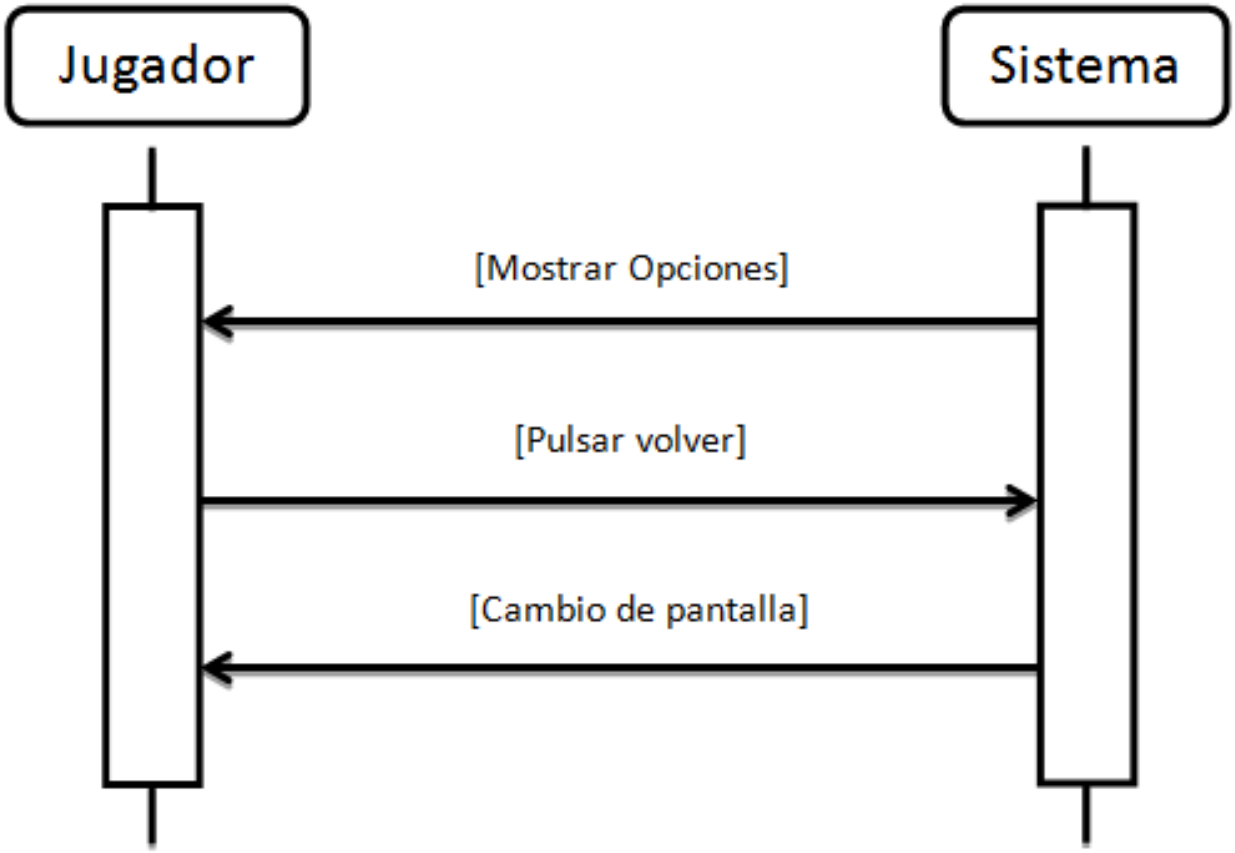

Figura 4.21: Diagrama de secuencias del sistema: Volver (Opciones)

Contrato de las operaciones

Operación: Mostrar Opciones
Responsabilidad: Mostramos las opciones.
Precondiciones: Se ha entrado en la pantalla de “Opciones”.
Postcondiciones: Se carga todo lo necesario para mostrar las opciones.

Operación: Pulsar volver
Responsabilidad: Seleccionar la opción “VOLVER”
Precondiciones: Se ha mostrado la pantalla de “Opciones”.
Postcondiciones: Se ha seleccionado la opción “VOLVER”

Operación: Cambio de pantalla
Responsabilidad: Se cambia de pantalla a la de “Menú principal”.
Precondiciones: Se ha seleccionado la opción “VOLVER”
Postcondiciones: Se ha cambiado a la pantalla de “Menú principal”.

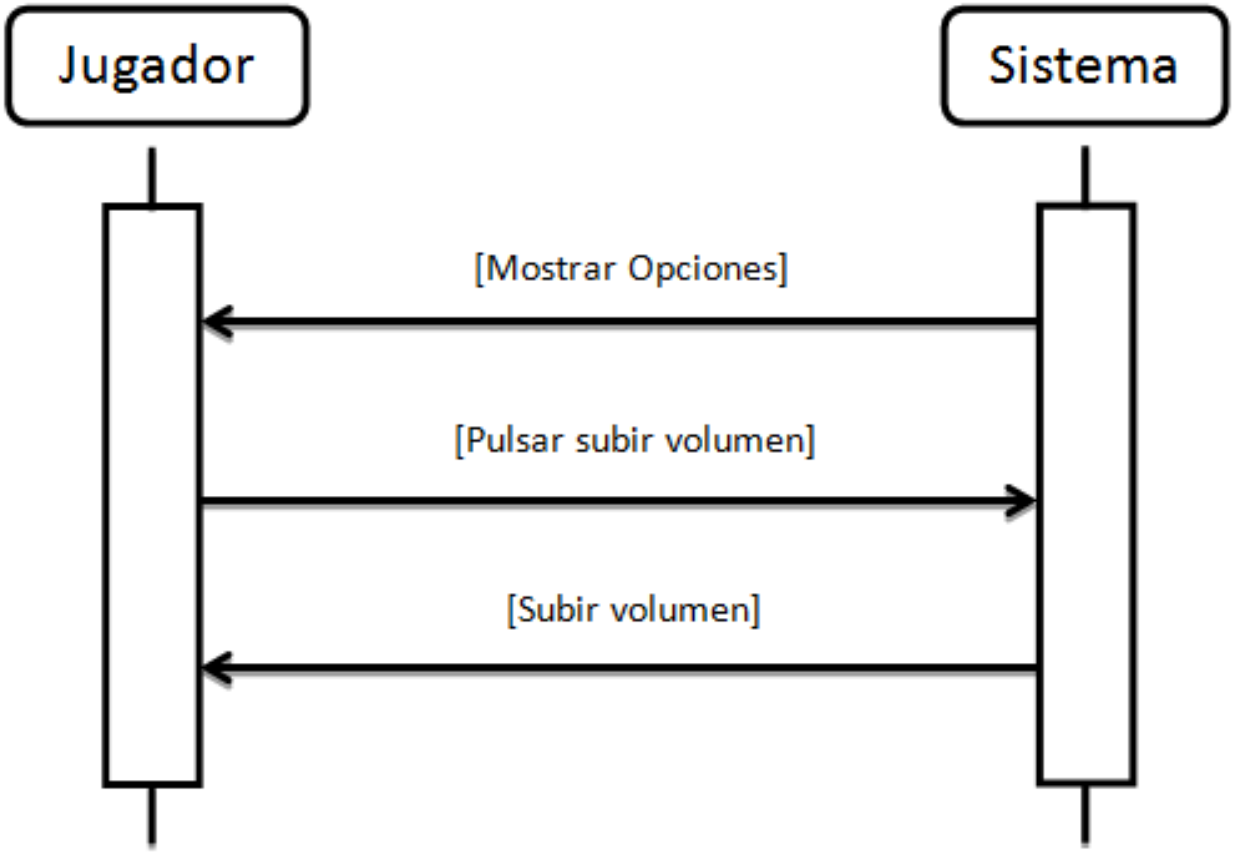

Figura 4.22: Diagrama de secuencias del sistema: Subir Volumen

Contrato de las operaciones

Operación: Mostrar Opciones
Responsabilidad: Mostramos las opciones.
Precondiciones: Se ha entrado en la pantalla de “Opciones”.
Postcondiciones: Se carga todo lo necesario para mostrar las opciones.

Operación: Pulsar subir volumen
Responsabilidad: Seleccionar la opción “SUBIR VOLUMEN”
Precondiciones: Se ha mostrado la pantalla de “Opciones”.
Postcondiciones: Se ha seleccionado la opción “SUBIR VOLUMEN”

Operación: Subir Volumen
Responsabilidad: Se sube el volumen un punto.
Precondiciones: Se ha seleccionado la opción “SUBIR VOLUMEN” y el volumen no está al máximo.
Postcondiciones: Se ha subido el volumen un punto.

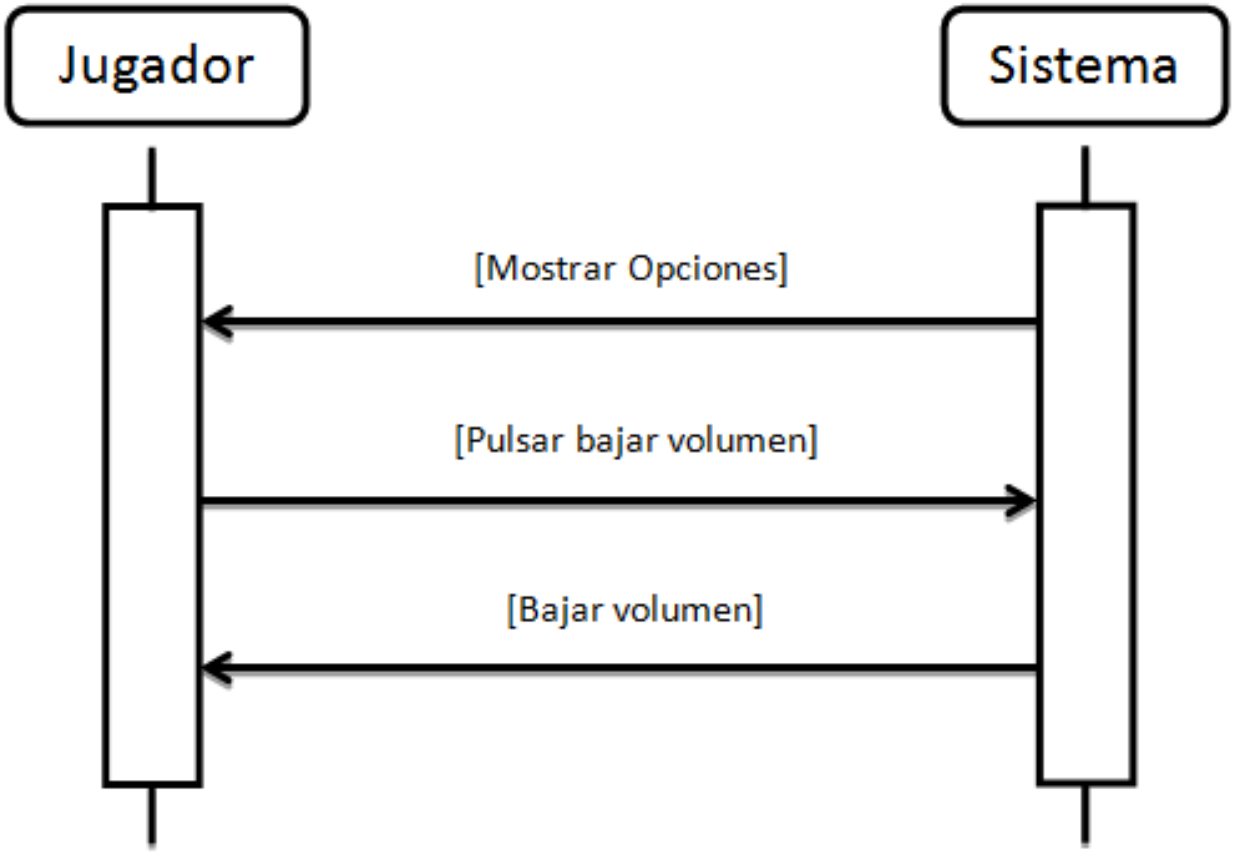

Figura 4.23: Diagrama de secuencias del sistema: Bajar Volumen

Contrato de las operaciones

Operación: Mostrar Opciones
Responsabilidad: Mostramos las opciones.
Precondiciones: Se ha entrado en la pantalla de “Opciones”.
Postcondiciones: Se carga todo lo necesario para mostrar las opciones.

Operación: Pulsar bajar volumen
Responsabilidad: Seleccionar la opción “BAJAR VOLUMEN”
Precondiciones: Se ha mostrado la pantalla de “Opciones”.
Postcondiciones: Se ha seleccionado la opción “BAJAR VOLUMEN”

Operación: Bajar Volumen
Responsabilidad: Se baja el volumen un punto.
Precondiciones: Se ha seleccionado la opción “BAJAR VOLUMEN” y el volumen no está al mínimo.
Postcondiciones: Se ha bajado el volumen un punto.

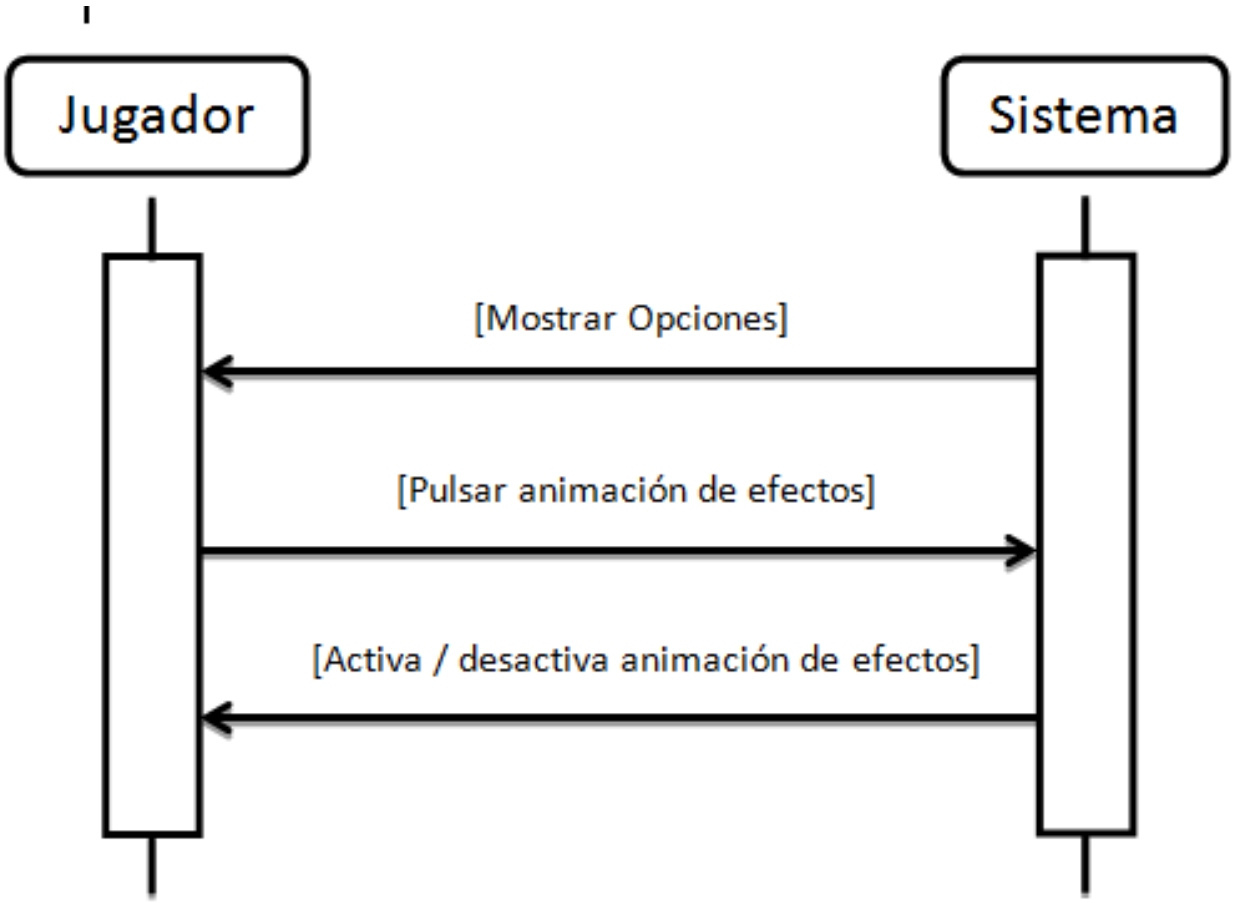

Figura 4.24: Diagrama de secuencias del sistema: Animaciones de efectos

Contrato de las operaciones

Operación: Mostrar Opciones
Responsabilidad: Mostramos las opciones.
Precondiciones: Se ha entrado en la pantalla de “Opciones”.
Postcondiciones: Se carga todo lo necesario para mostrar las opciones.

Operación: Pulsar animación de efectos
Responsabilidad: Seleccionar la opción “ANIMACION DE EFEECTOS”
Precondiciones: Se ha mostrado la pantalla de “Opciones”.
Postcondiciones: Se ha seleccionado la opción “ANIMACION DE EFECTOS”

Operación: Activa/Desactiva animación de efectos
Responsabilidad: Se activa/desactiva las animaciones de los efectos.
Precondiciones: Se ha seleccionado la opción “ANIMACION DE EFECTOS”.
Postcondiciones: Se activa/desactiva las animaciones de los efectos.

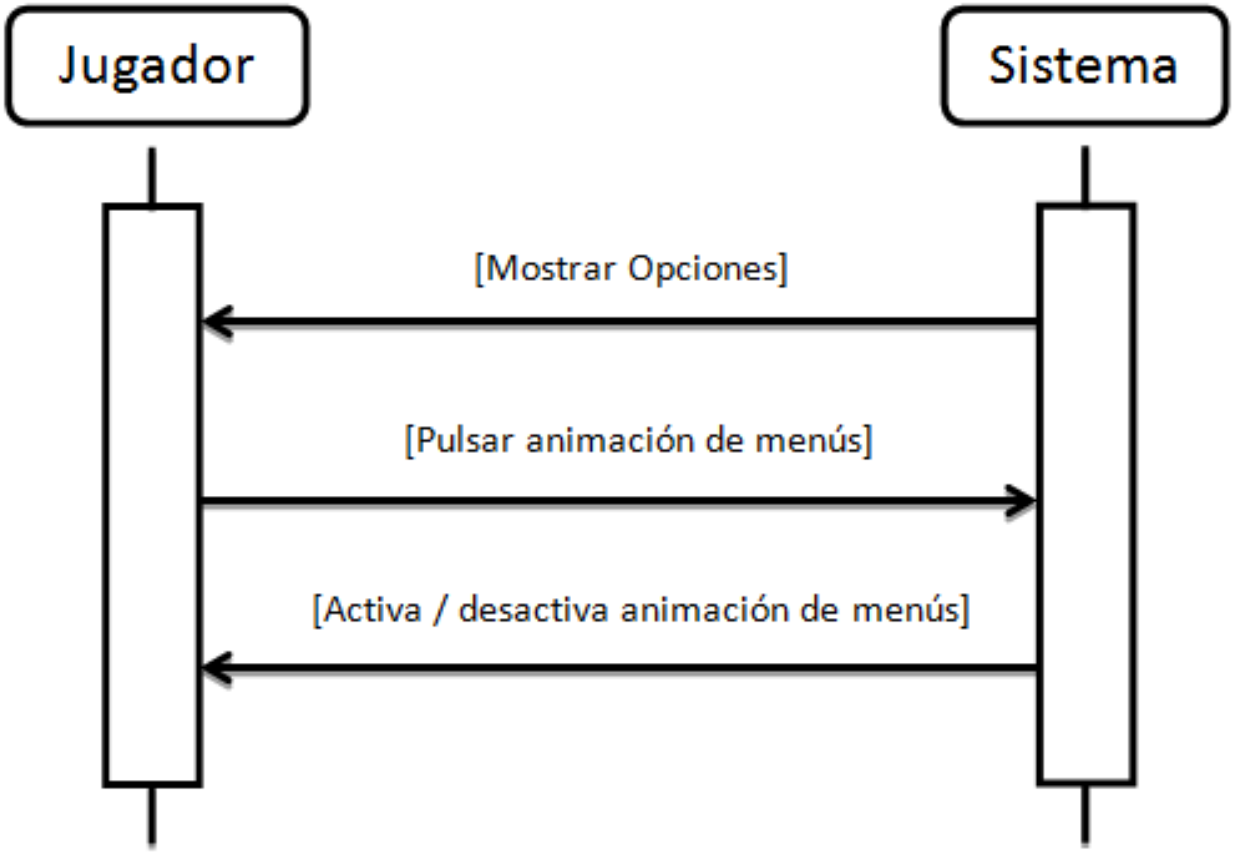

Figura 4.25: Diagrama de secuencias del sistema: Animaciones de menús

Contrato de las operaciones

Operación: Mostrar Opciones
Responsabilidad: Mostramos las opciones.
Precondiciones: Se ha entrado en la pantalla de “Opciones”.
Postcondiciones: Se carga todo lo necesario para mostrar las opciones.

Operación: Pulsar animación de menús
Responsabilidad: Seleccionar la opción “ANIMACION DE MENUS”
Precondiciones: Se ha mostrado la pantalla de “Opciones”.
Postcondiciones: Se ha seleccionado la opción “ANIMACION DE MENUS”

Operación: Activa/Desactiva animación de menús
Responsabilidad: Se activa/desactiva las animaciones de los menús.
Precondiciones: Se ha seleccionado la opción “ANIMACION DE MENUS”.
Postcondiciones: Se activa/desactiva las animaciones de los menús.

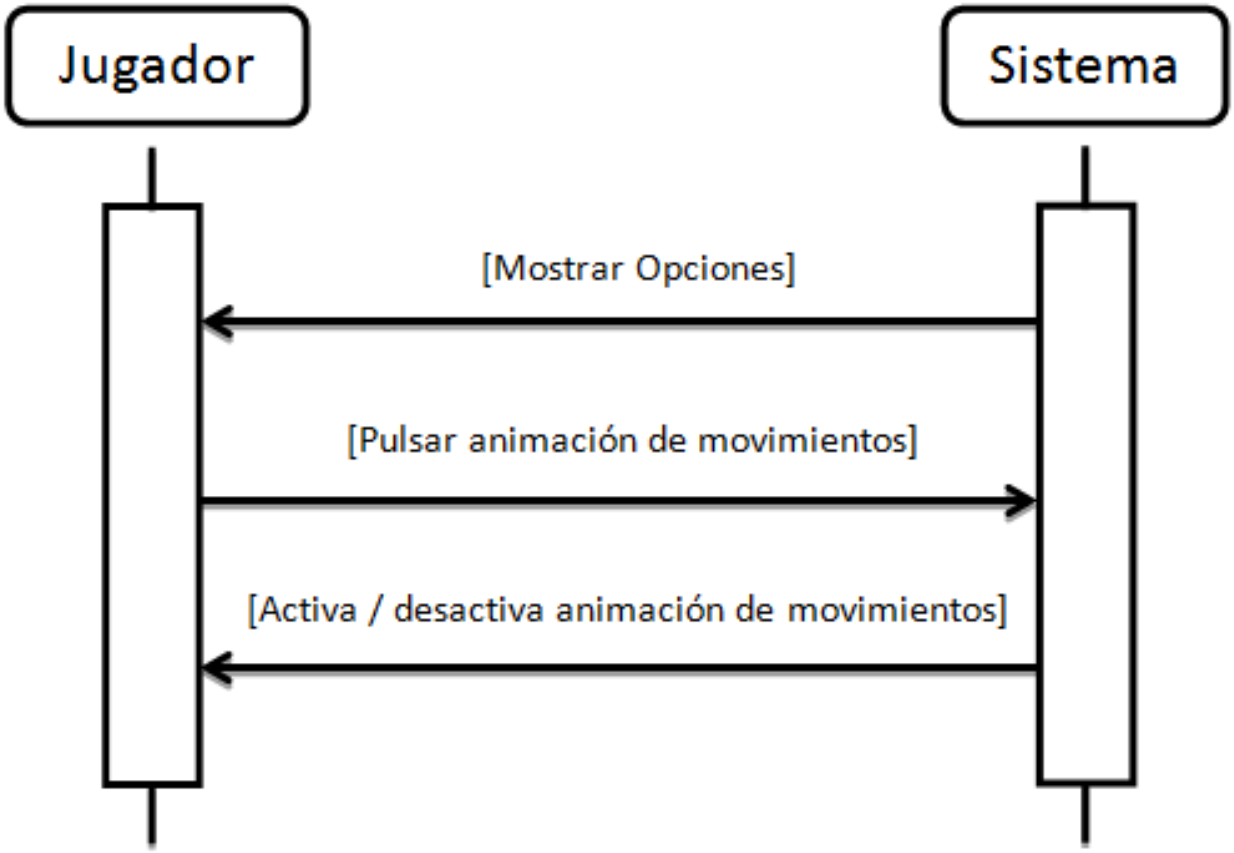

Figura 4.26: Diagrama de secuencias del sistema: Animaciones de movimientos

Contrato de las operaciones

Operación: Mostrar Opciones
Responsabilidad: Mostramos las opciones.
Precondiciones: Se ha entrado en la pantalla de "Opciones".
Postcondiciones: Se carga todo lo necesario para mostrar las opciones.

Operación: Pulsar animación de movimientos
Responsabilidad: Seleccionar la opción "ANIMACION DE MOVIMIENTOS"
Precondiciones: Se ha mostrado la pantalla de "Opciones".
Postcondiciones: Se ha seleccionado la opción "ANIMACION DE MOVIMIENTOS"

Operación: Activa/Desactiva animación de movimientos
Responsabilidad: Se activa/desactiva las animaciones de los movimientos.
Precondiciones: Se ha seleccionado la opción "ANIMACION DE MOVIMIENTOS".
Postcondiciones: Se activa/desactiva las animaciones de los movimientos.

4.3.6. Juego

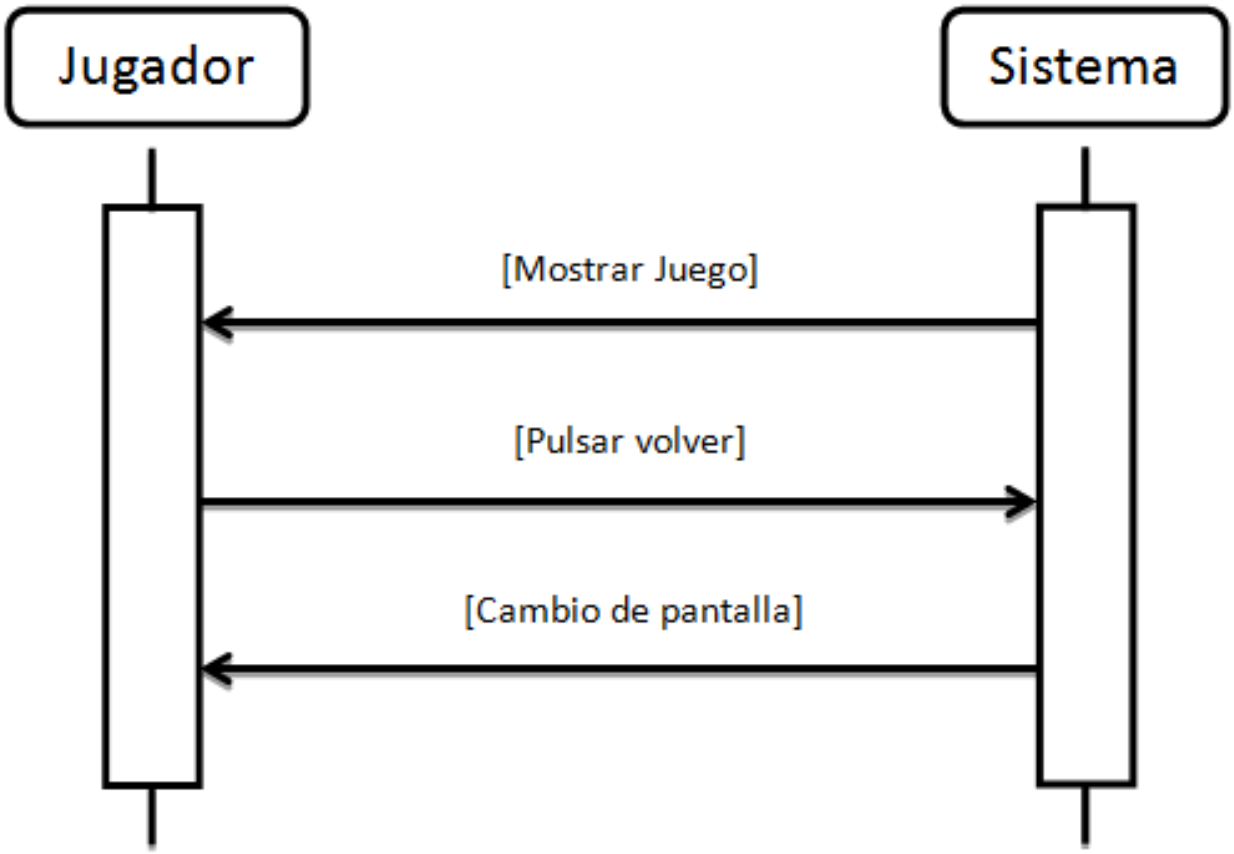

Figura 4.27: Diagrama de secuencias del sistema: Volver (Juego)

Contrato de las operaciones

Operación: Mostrar Juego
Responsabilidad: Mostramos las opciones para jugar.
Precondiciones: Se ha entrado en la pantalla de “Juego”.
Postcondiciones: Se carga todo lo necesario para mostrar las opciones para jugar.

Operación: Pulsar volver
Responsabilidad: Seleccionar la opción “VOLVER”
Precondiciones: Se ha mostrado la pantalla de “Juego”.
Postcondiciones: Se ha seleccionado la opción “VOLVER”

Operación: Cambio de pantalla
Responsabilidad: Se cambia de pantalla a la de “Menú principal”.
Precondiciones: Se ha seleccionado la opción “VOLVER”
Postcondiciones: Se ha cambiado a la pantalla de “Menú principal”.

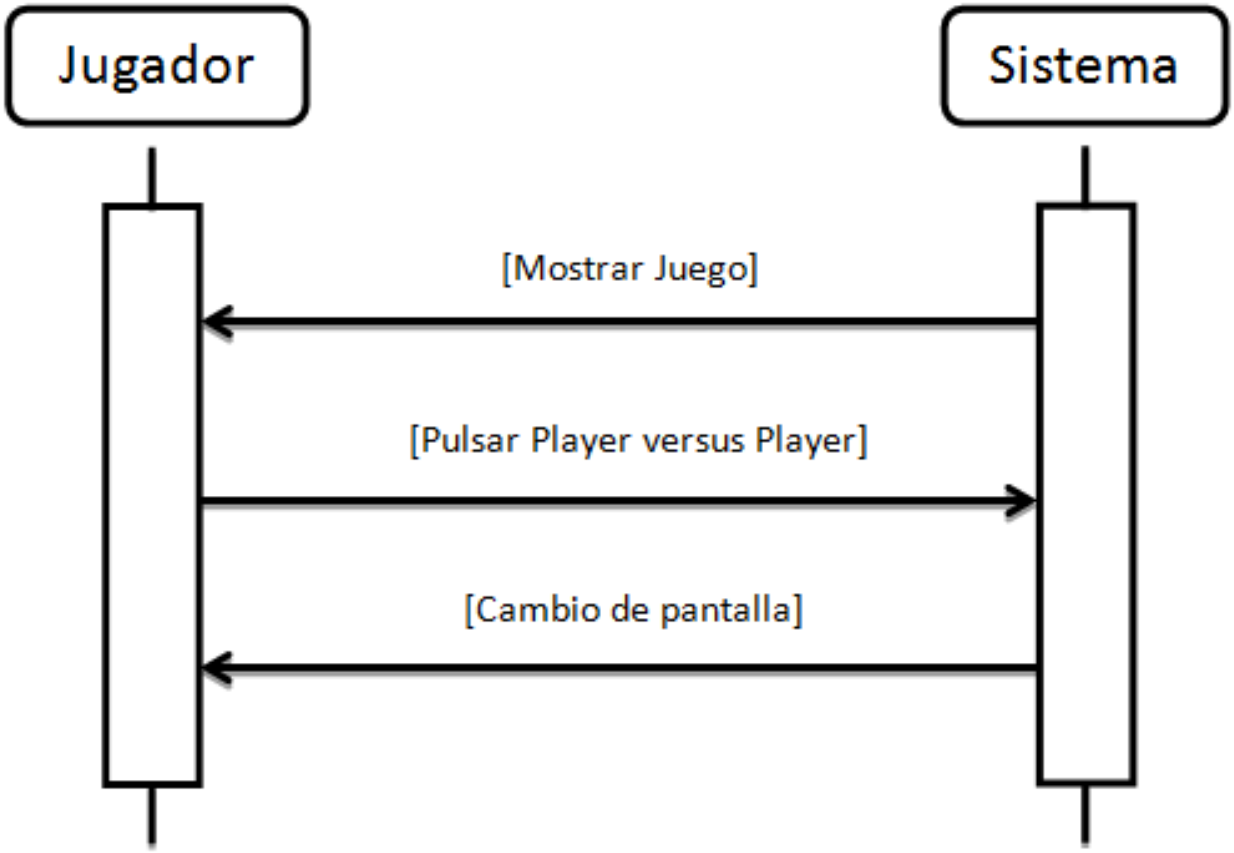

Figura 4.28: Diagrama de secuencias del sistema: Player versus Player

Contrato de las operaciones

Operación: Mostrar Juego
Responsabilidad: Mostramos las opciones para jugar.
Precondiciones: Se ha entrado en la pantalla de “Juego”.
Postcondiciones: Se carga todo lo necesario para mostrar las opciones para jugar.

Operación: Pulsar Player versus Player
Responsabilidad: Seleccionar la opción “PLAYER VERSUS PLAYER”
Precondiciones: Se ha mostrado la pantalla de “Juego”.
Postcondiciones: Se ha seleccionado la opción “PLAYER VERSUS PLAYER”

Operación: Cambio de pantalla
Responsabilidad: Se cambia de pantalla a la de “Selección”.
Precondiciones: Se ha seleccionado la opción “PLAYER VERSUS PLAYER”
Postcondiciones: Se ha cambiado a la pantalla de “Selección”.

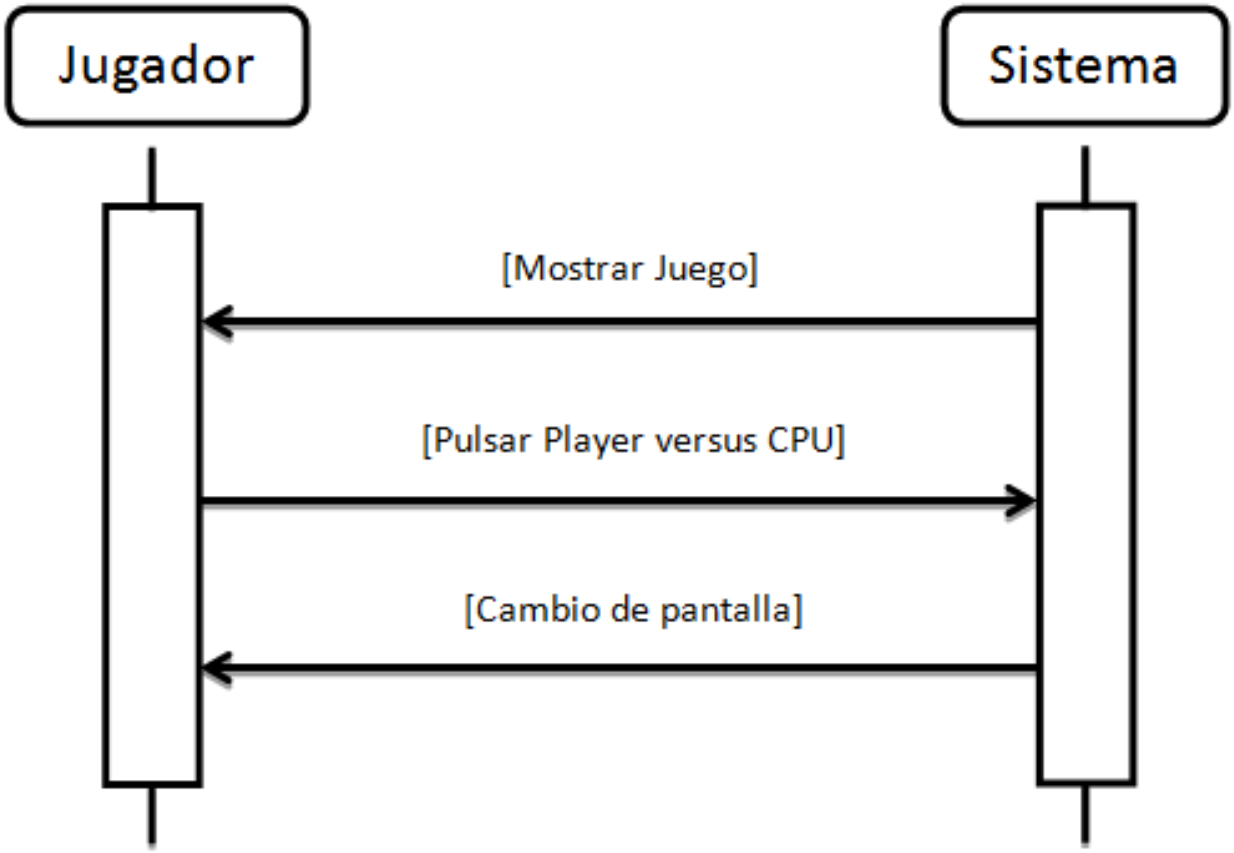

Figura 4.29: Diagrama de secuencias del sistema: Player versus CPU

Contrato de las operaciones

Operación: Mostrar Juego
Responsabilidad: Mostramos las opciones para jugar.
Precondiciones: Se ha entrado en la pantalla de “Juego”.
Postcondiciones: Se carga todo lo necesario para mostrar las opciones para jugar.

Operación: Pulsar Player versus CPU
Responsabilidad: Seleccionar la opción “PLAYER VERSUS CPU”
Precondiciones: Se ha mostrado la pantalla de “Juego”.
Postcondiciones: Se ha seleccionado la opción “PLAYER VERSUS CPU”

Operación: Cambio de pantalla
Responsabilidad: Se cambia de pantalla a la de “Selección”.
Precondiciones: Se ha seleccionado la opción “PLAYER VERSUS CPU”
Postcondiciones: Se ha cambiado a la pantalla de “Selección”.

4.3.7. Selección

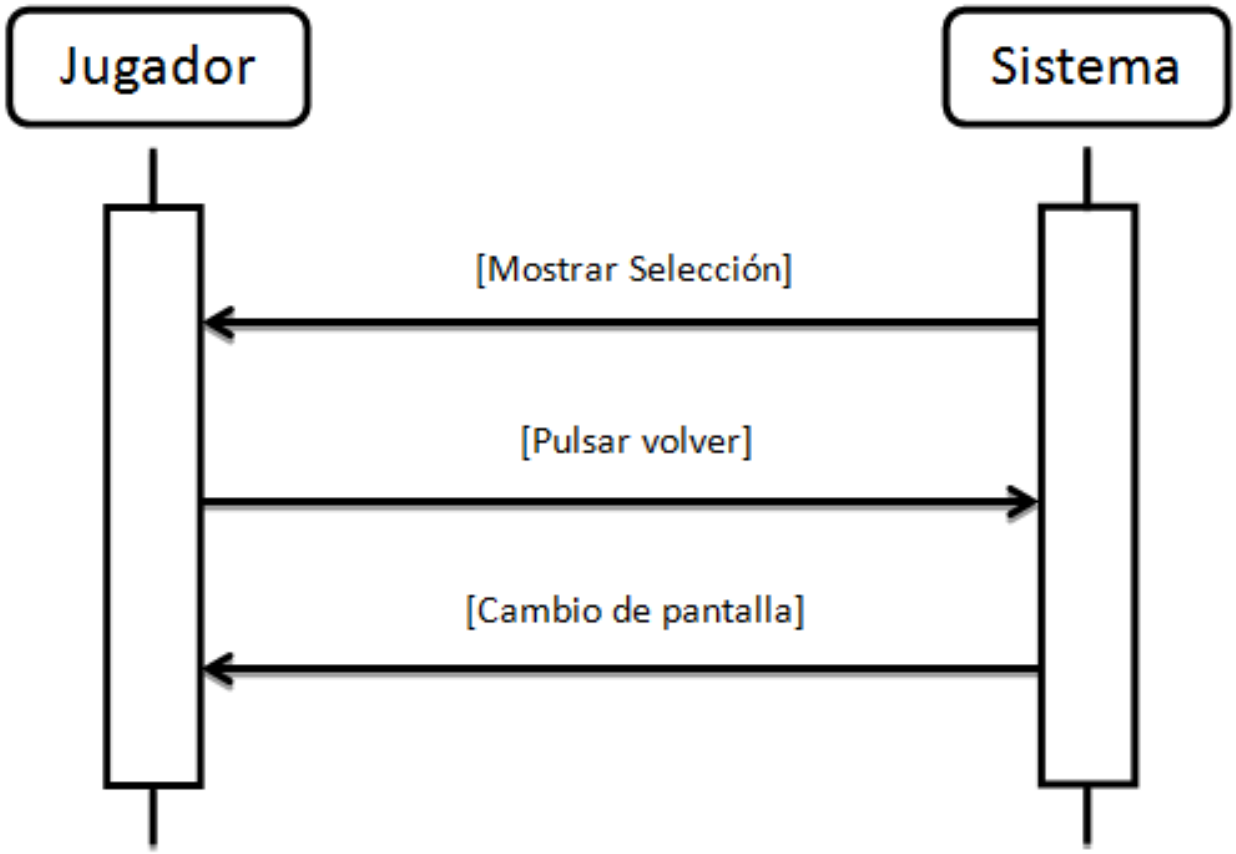

Figura 4.30: Diagrama de secuencias del sistema: Volver (Selección)

Contrato de las operaciones

Operación: Mostrar Selección
Responsabilidad: Mostramos las opciones de selección para jugar
Precondiciones: Se ha entrado en la pantalla de “Selección”.
Postcondiciones: Se carga todo lo necesario para mostrar las opciones de selección para jugar.

Operación: Pulsar volver
Responsabilidad: Seleccionar la opción “VOLVER”
Precondiciones: Se ha mostrado la pantalla de “Selección”.
Postcondiciones: Se ha seleccionado la opción “VOLVER”

Operación: Cambio de pantalla
Responsabilidad: Se cambia de pantalla a la de “Juego”.
Precondiciones: Se ha seleccionado la opción “VOLVER”
Postcondiciones: Se ha cambiado a la pantalla de “Juego”.

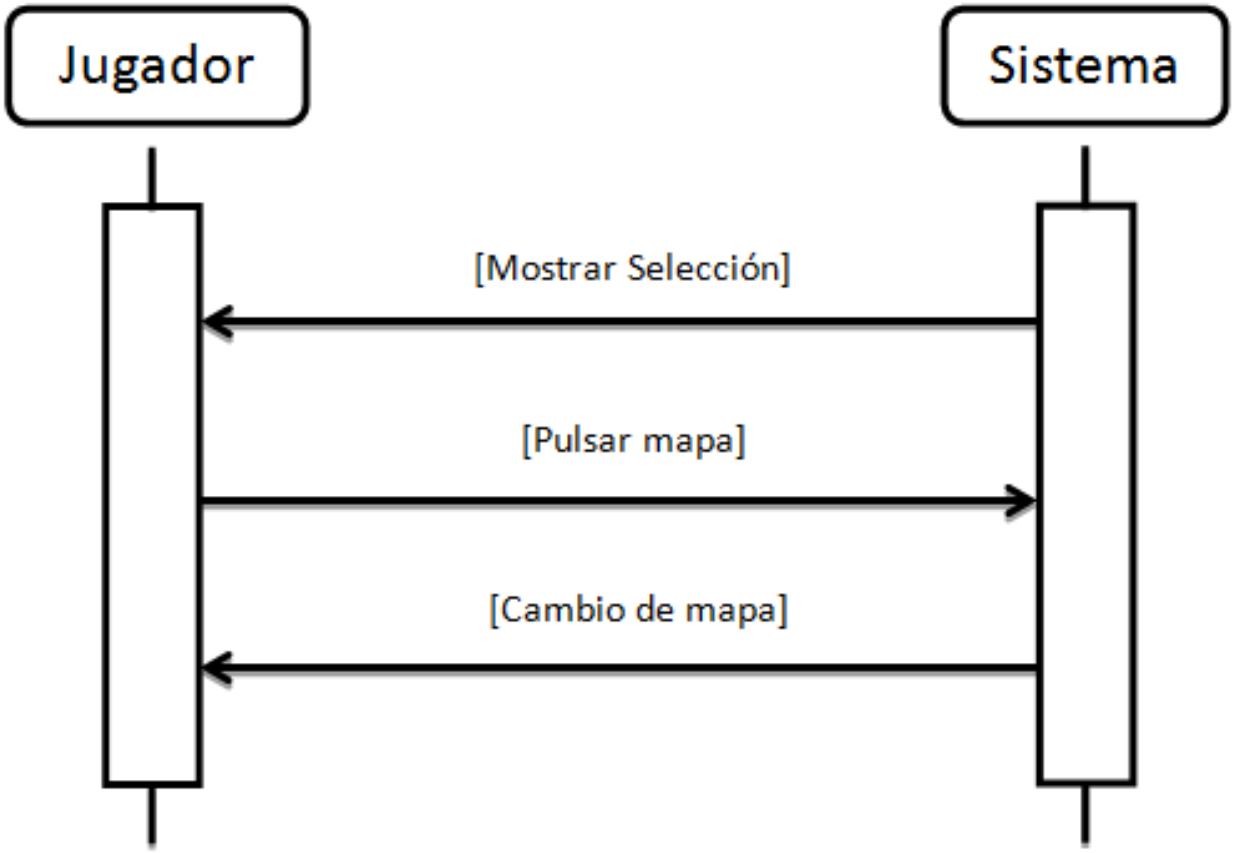

Figura 4.31: Diagrama de secuencias del sistema: Mapa

Contrato de las operaciones

Operación: Mostrar Selección
Responsabilidad: Mostramos las opciones de selección para jugar.
Precondiciones: Se ha entrado en la pantalla de "Selección".
Postcondiciones: Se carga todo lo necesario para mostrar las opciones de selección para jugar.

Operación: Pulsar mapa
Responsabilidad: Seleccionar un nuevo mapa.
Precondiciones: Se ha mostrado la pantalla de "Selección".
Postcondiciones: Se ha seleccionado un nuevo mapa.

Operación: Cambio de mapa
Responsabilidad: Se cambia de mapa.
Precondiciones: Se ha seleccionado un mapa nuevo.
Postcondiciones: Se ha cambiado de mapa.

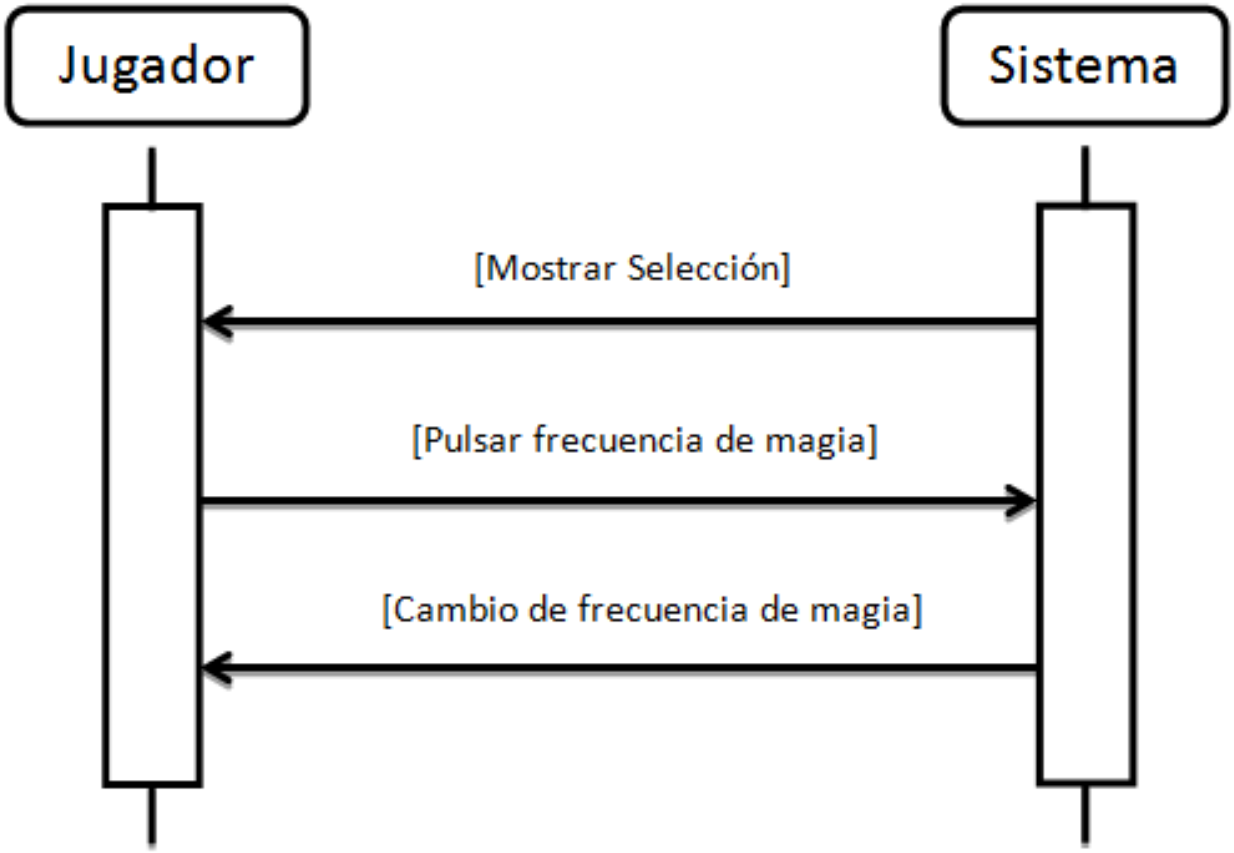

Figura 4.32: Diagrama de secuencias del sistema: Frecuencia de magia

Contrato de las operaciones

Operación: Mostrar Selección
Responsabilidad: Mostramos las opciones de selección para jugar.
Precondiciones: Se ha entrado en la pantalla de "Selección".
Postcondiciones: Se carga todo lo necesario para mostrar las opciones de selección para jugar.

Operación: Pulsar frecuencia de magia
Responsabilidad: Seleccionar una nueva frecuencia de magia.
Precondiciones: Se ha mostrado la pantalla de "Selección".
Postcondiciones: Se ha seleccionado una nueva frecuencia de magia.

Operación: Cambio de frecuencia de magia
Responsabilidad: Se cambia de frecuencia de magia.
Precondiciones: Se ha seleccionado una frecuencia de magia nueva.
Postcondiciones: Se ha cambiado de frecuencia de magia.

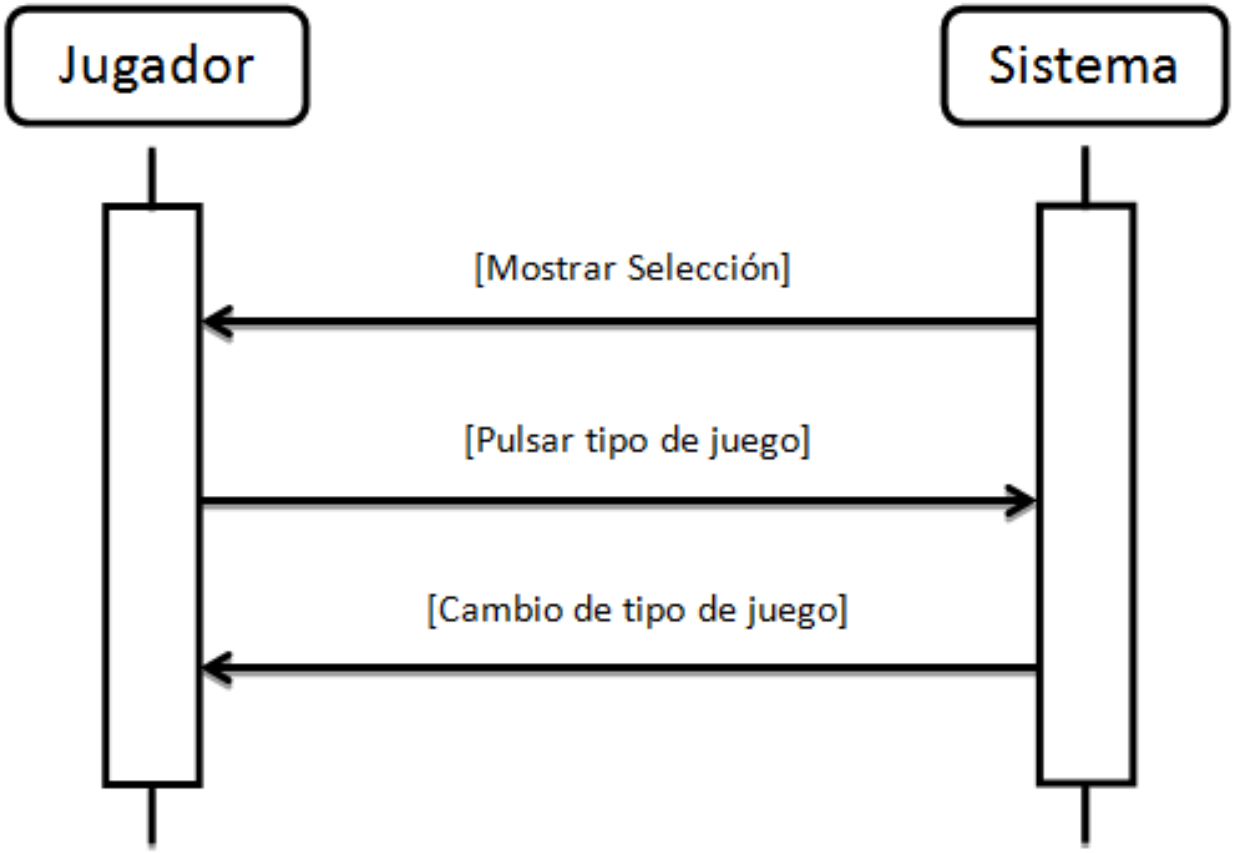

Figura 4.33: Diagrama de secuencias del sistema: Tipo de juego

Contrato de las operaciones

Operación: Mostrar Selección
Responsabilidad: Mostramos las opciones de selección para jugar.
Precondiciones: Se ha entrado en la pantalla de "Selección".
Postcondiciones: Se carga todo lo necesario para mostrar las opciones de selección para jugar.

Operación: Pulsar tipo de juego
Responsabilidad: Seleccionar una nuevo tipo de juego.
Precondiciones: Se ha mostrado la pantalla de "Selección".
Postcondiciones: Se ha seleccionado un nuevo tipo de juego.

Operación: Cambio de tipo de juego
Responsabilidad: Se cambia de tipo de juego.
Precondiciones: Se ha seleccionado un tipo de juego nuevo.
Postcondiciones: Se ha cambiado de tipo de juego.

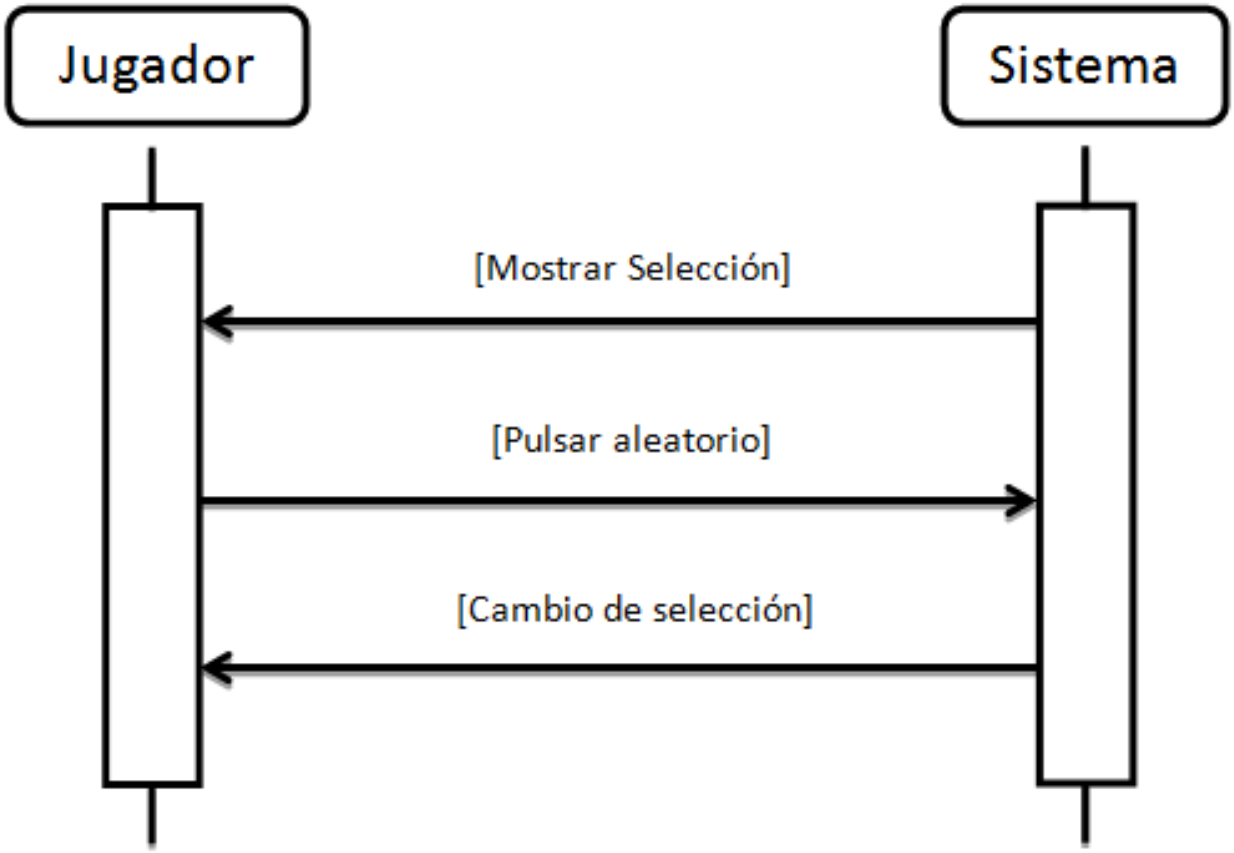

Figura 4.34: Diagrama de secuencias del sistema: Aleatorio

Contrato de las operaciones

Operación: Mostrar Selección
Responsabilidad: Mostramos las opciones de selección para jugar.
Precondiciones: Se ha entrado en la pantalla de "Selección".
Postcondiciones: Se carga todo lo necesario para mostrar las opciones de selección para jugar.

Operación: Pulsar Aleatorio
Responsabilidad: Seleccionar la opción "ALEATORIO"
Precondiciones: Se ha mostrado la pantalla de "Selección".
Postcondiciones: Se ha seleccionado la opción "ALEATORIO"

Operación: Cambio de selección
Responsabilidad: Se cambia de todas las opciones.
Precondiciones: Se ha seleccionado la opción de "ALEATORIO"
Postcondiciones: Se ha cambiado de todas las opciones.

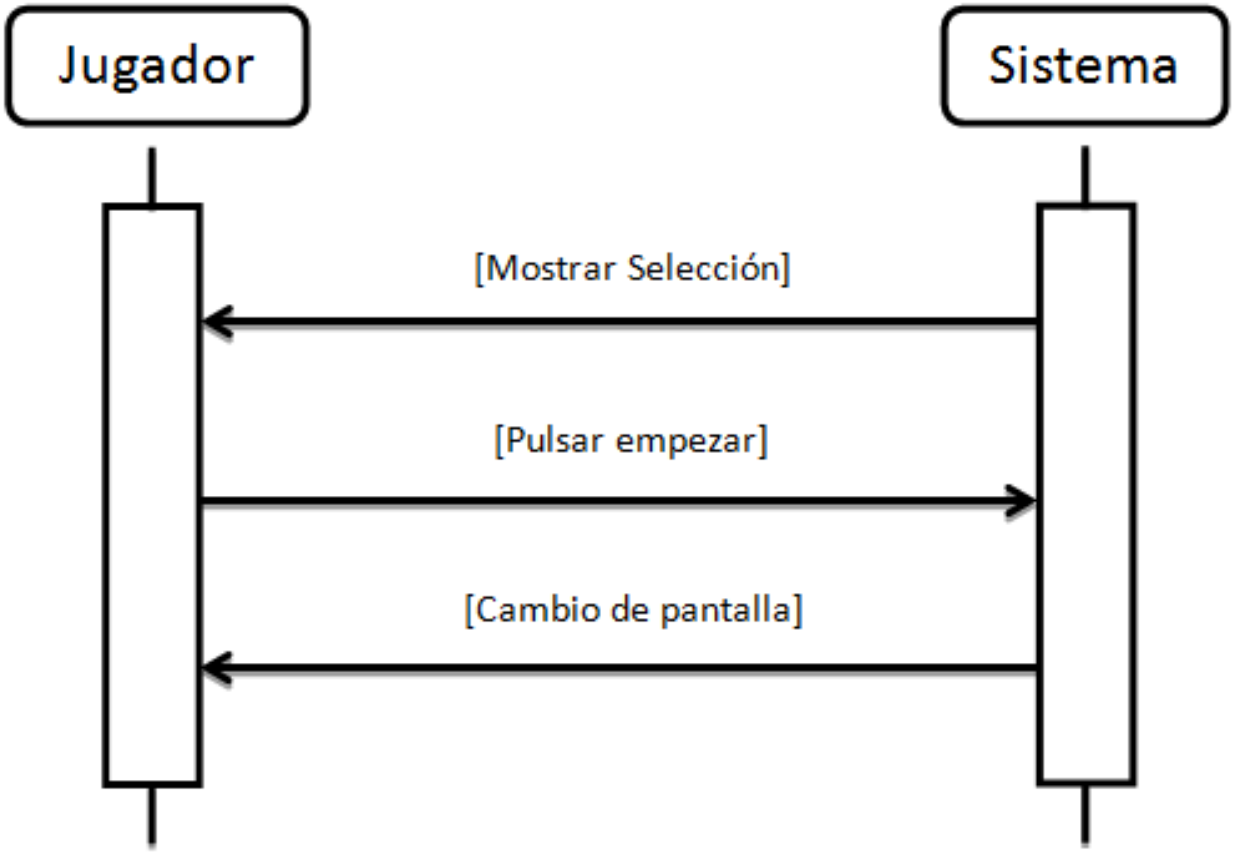

Figura 4.35: Diagrama de secuencias del sistema: Empezar

Contrato de las operaciones

Operación: Mostrar Selección
Responsabilidad: Mostramos las opciones de selección para jugar.
Precondiciones: Se ha entrado en la pantalla de “Selección”.
Postcondiciones: Se carga todo lo necesario para mostrar las opciones de selección para jugar.

Operación: Pulsar Empezar
Responsabilidad: Seleccionar la opción “EMPEZAR”
Precondiciones: Se ha mostrado la pantalla de “Selección”.
Postcondiciones: Se ha seleccionado la opción “EMPEZAR”

Operación: Cambio de pantalla
Responsabilidad: Se cambia de pantalla a la de “Jugando”
Precondiciones: Se ha seleccionado la opción de “EMPEZAR”
Postcondiciones: Se ha cambiado de pantalla a “Jugando”.

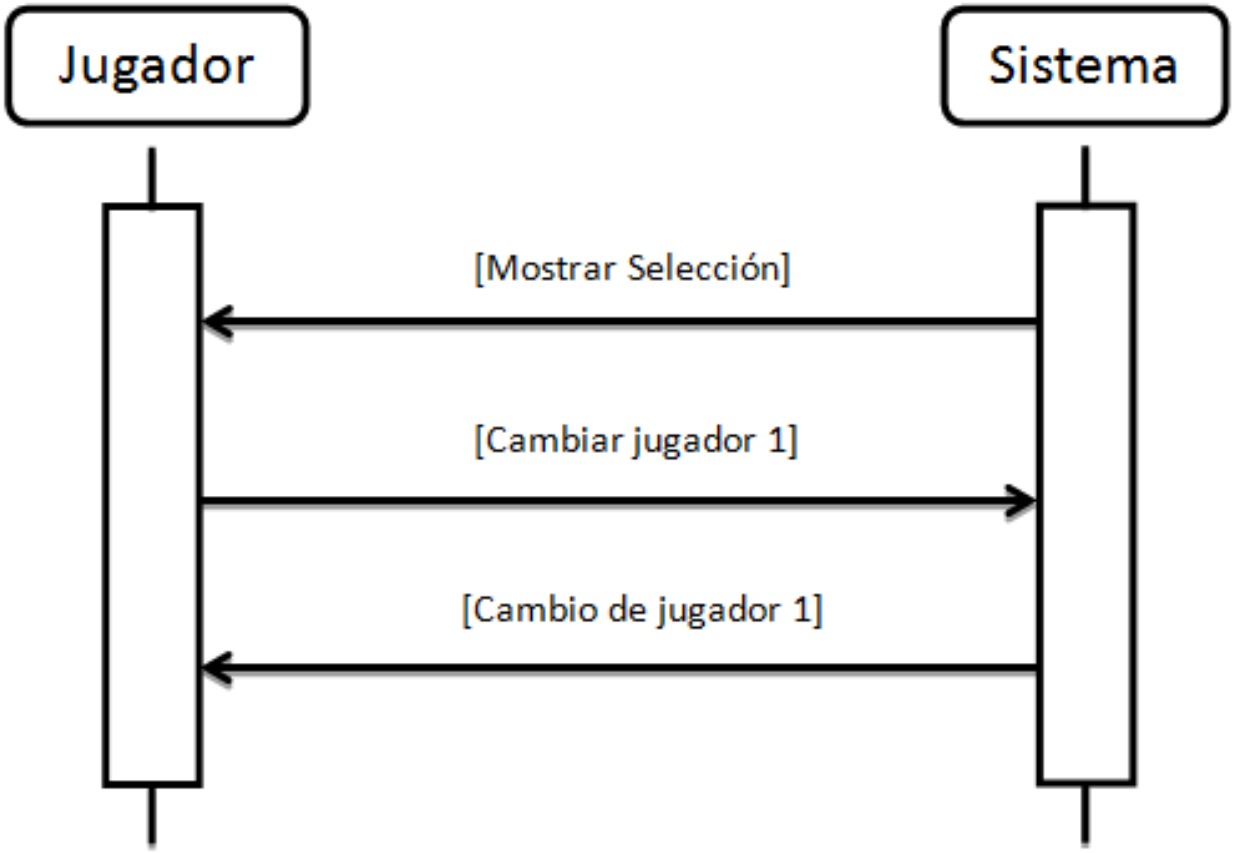

Figura 4.36: Diagrama de secuencias del sistema: Jugador 1

Contrato de las operaciones

Operación: Mostrar Selección
Responsabilidad: Mostramos las opciones de selección para jugar.
Precondiciones: Se ha entrado en la pantalla de "Selección".
Postcondiciones: Se carga todo lo necesario para mostrar las opciones de selección para jugar.

Operación: Cambiar Jugador 1
Responsabilidad: Seleccionar las nuevas características del Jugador 1.
Precondiciones: Se ha mostrado la pantalla de "Selección".
Postcondiciones: Se ha seleccionado las nuevas características del Jugador 1.

Operación: Cambio de Jugador 1
Responsabilidad: El sistema cambia las características del Jugador 1.
Precondiciones: Se ha seleccionado las características nuevas del Jugador 1.
Postcondiciones: El sistema ha cambiado las características del Jugador 1.

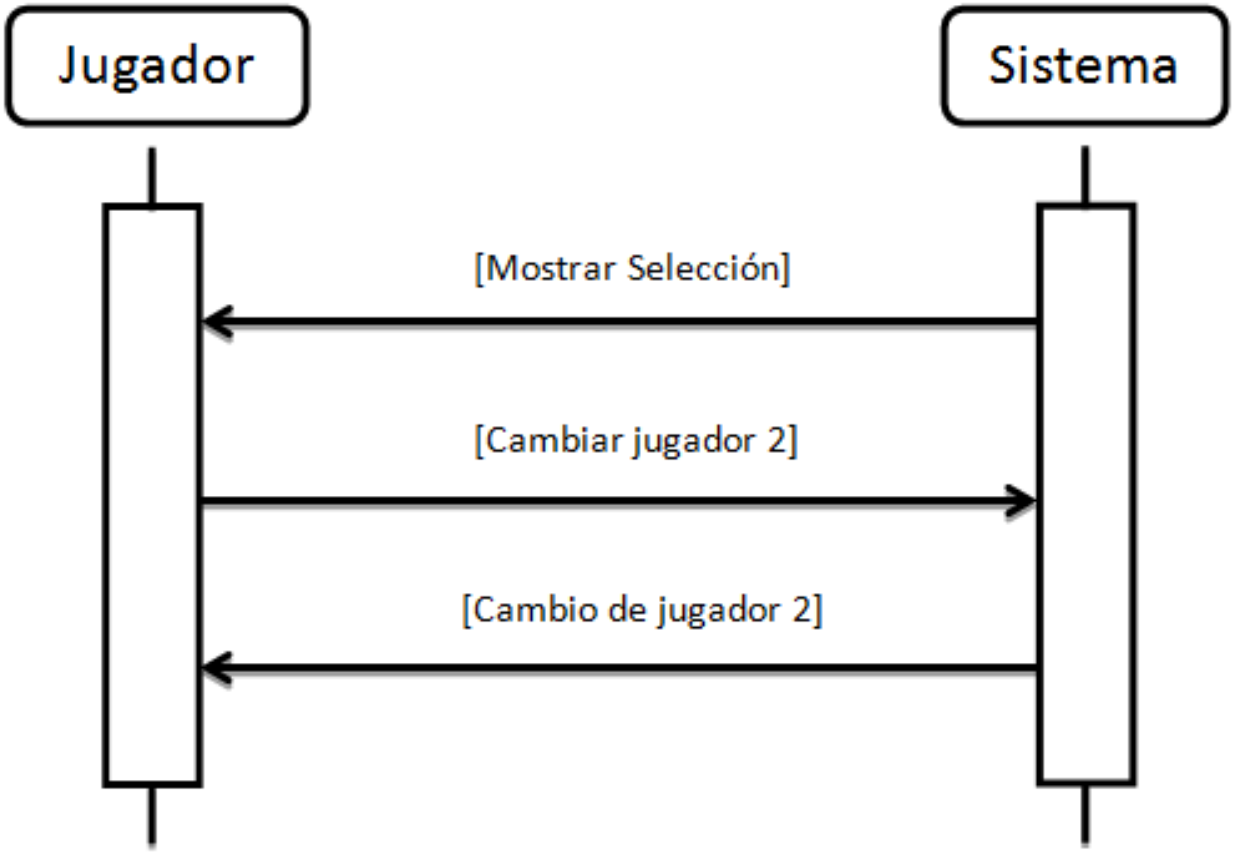

Figura 4.37: Diagrama de secuencias del sistema: Jugador 2

Contrato de las operaciones

Operación: Mostrar Selección
Responsabilidad: Mostramos las opciones de selección para jugar.
Precondiciones: Se ha entrado en la pantalla de "Selección" eligiendo la opción de "PLAYER VERSUS PLAYER".
Postcondiciones: Se carga todo lo necesario para mostrar las opciones de selección para jugar.

Operación: Cambiar Jugador 2
Responsabilidad: Seleccionar las nuevas características del Jugador 2.
Precondiciones: Se ha mostrado la pantalla de "Selección".
Postcondiciones: Se ha seleccionado las nuevas características del Jugador 2.

Operación: Cambio de Jugador 2
Responsabilidad: El sistema cambia las características del Jugador 2.
Precondiciones: Se ha seleccionado las características nuevas del Jugador 2.
Postcondiciones: El sistema ha cambiado las características del Jugador 2.

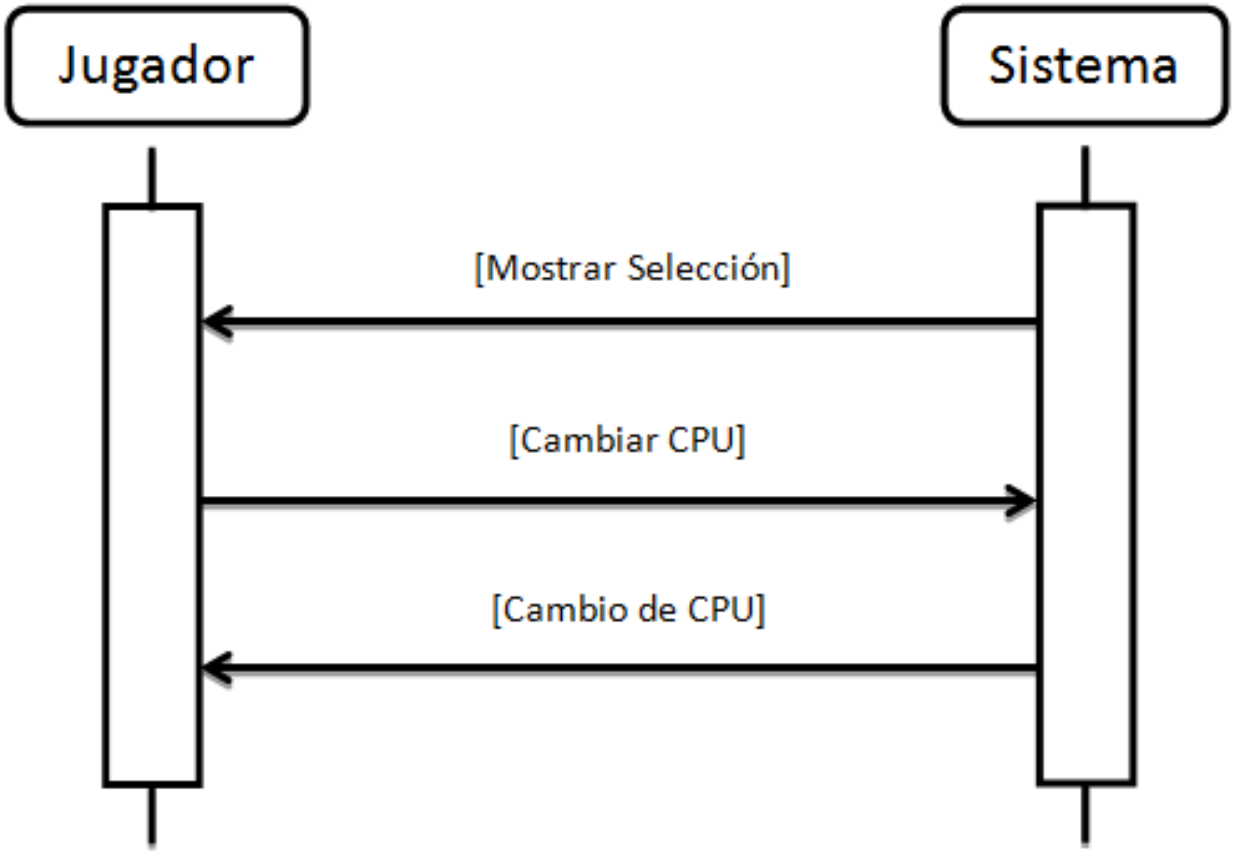

Figura 4.38: Diagrama de secuencias del sistema: CPU

Contrato de las operaciones

Operación: Mostrar Selección
Responsabilidad: Mostramos las opciones de selección para jugar.
Precondiciones: Se ha entrado en la pantalla de "Selección" eligiendo la opción de "PLAYER VERSUS CPU".
Postcondiciones: Se carga todo lo necesario para mostrar las opciones de selección para jugar.

Operación: Cambiar CPU
Responsabilidad: Seleccionar las nuevas características de la CPU.
Precondiciones: Se ha mostrado la pantalla de "Selección".
Postcondiciones: Se ha seleccionado las nuevas características dela CPU.

Operación: Cambio de CPU
Responsabilidad: El sistema cambia las características de la CPU.
Precondiciones: Se ha seleccionado las características nuevas de la CPU.
Postcondiciones: El sistema ha cambiado las características de la CPU.

4.3.8. Jugando

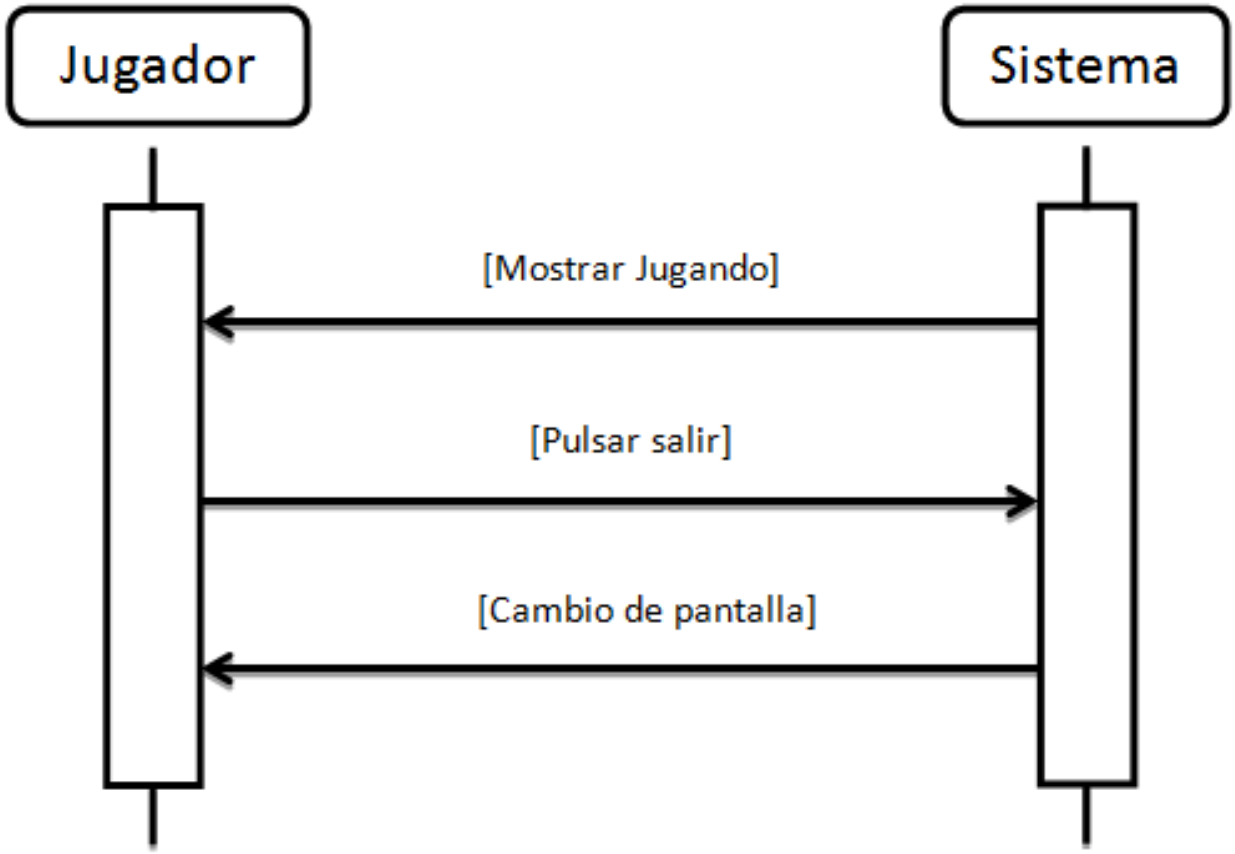

Figura 4.39: Diagrama de secuencias del sistema: Salir (Jugando)

Contrato de las operaciones

Operación: Mostrar Jugando
Responsabilidad: Mostrar la pantalla “Jugando” con las distintas opciones a elegir.
Precondiciones: Se ha entrado en la pantalla de “Jugando”.
Postcondiciones: Se carga todo lo necesario para mostrar la pantalla de “Jugando”.

Operación: Selección Salir
Responsabilidad: Seleccionar la opción “SALIR”
Precondiciones: Se ha mostrado la pantalla “Jugando”.
Postcondiciones: Se ha seleccionado la opción “SALIR”

Operación: Cambio de pantalla
Responsabilidad: Se cambia de pantalla a “Selección”.
Precondiciones: Se ha seleccionado la opción “SALIR”.
Postcondiciones: Se ha cambiado de pantalla a “Selección”.

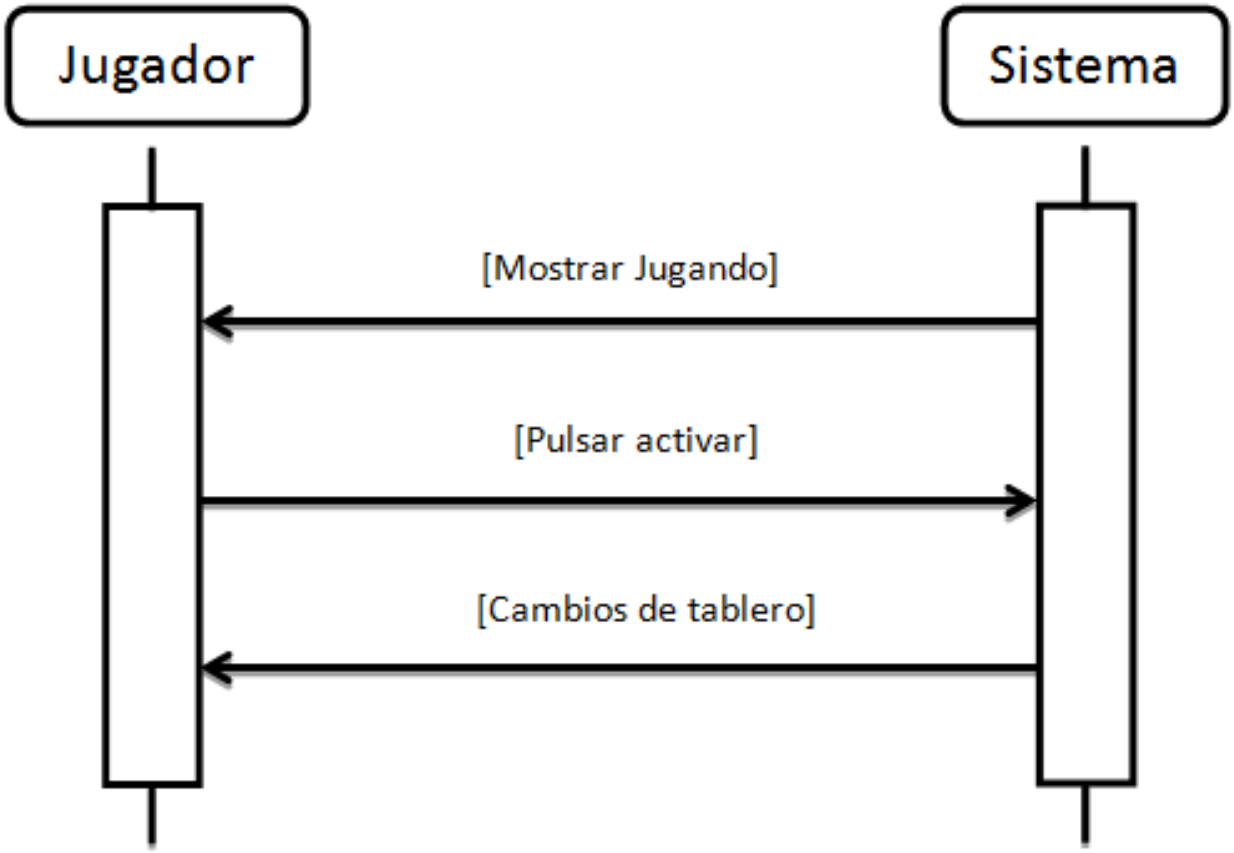

Figura 4.40: Diagrama de secuencias del sistema: Activar

Contrato de las operaciones

Operación: Mostrar Jugando
Responsabilidad: Mostrar la pantalla “Jugando” con las distintas opciones a elegir.
Precondiciones: Se ha entrado en la pantalla de “Jugando”.
Postcondiciones: Se carga todo lo necesario para mostrar la pantalla de “Jugando”.

Operación: Pulsar Activar
Responsabilidad: Seleccionar la opción “ACTIVAR”
Precondiciones: Se ha mostrado la pantalla de “Jugando”.
Postcondiciones: Se ha seleccionado la opción “ACTIVAR”

Operación: Cambio de tablero
Responsabilidad: Se cambia el tablero de juego.
Precondiciones: Se ha seleccionado la opción de “ACTIVAR”
Postcondiciones: Se ha cambiado el tablero de juego.

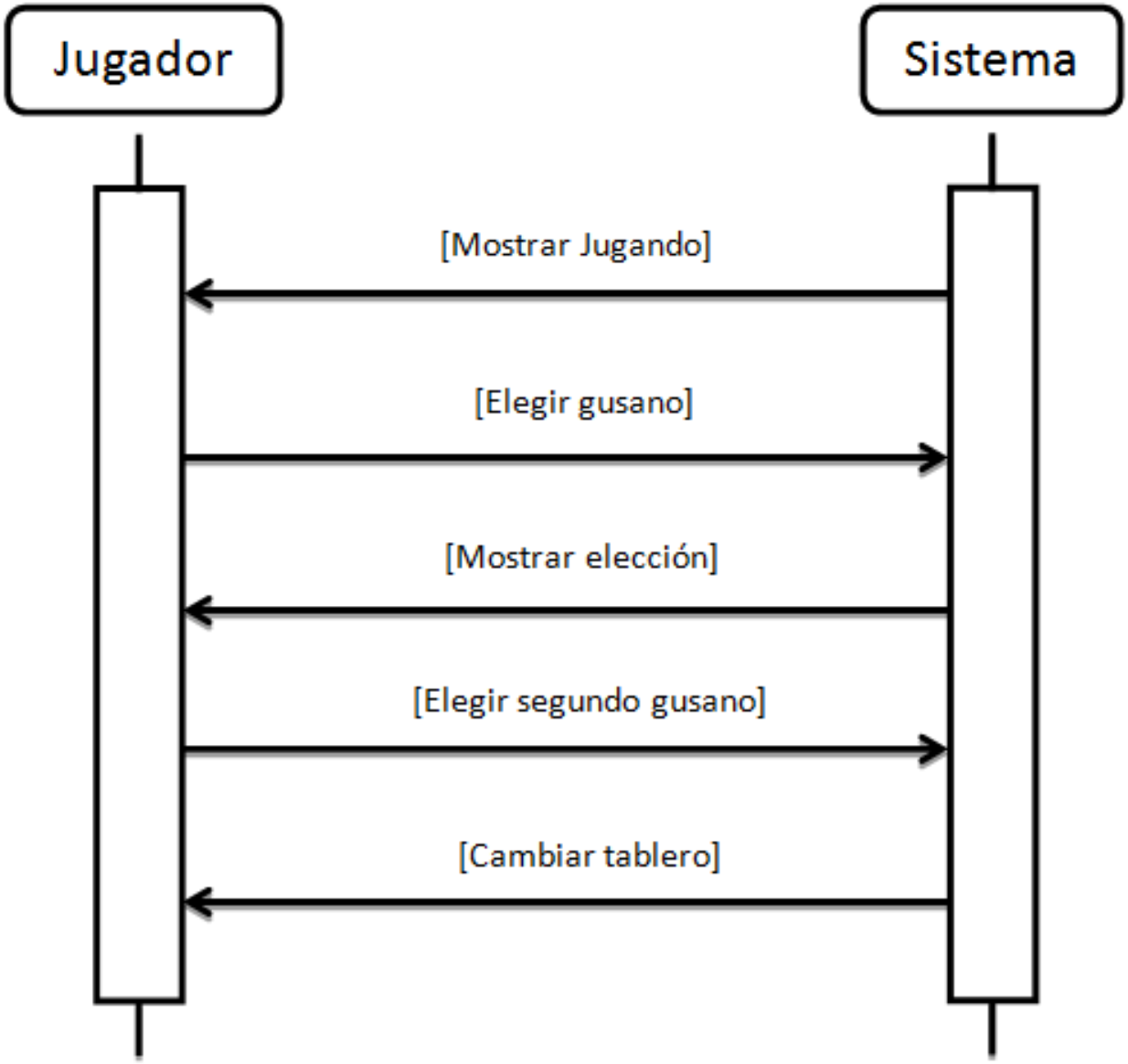

Figura 4.41: Diagrama de secuencias del sistema: Jugando

Contrato de las operaciones

Operación: Mostrar Jugando
Responsabilidad: Mostrar la pantalla “Jugando” con las distintas opciones a elegir.
Precondiciones: Se ha entrado en la pantalla de “Jugando”.
Postcondiciones: Se carga todo lo necesario para mostrar la pantalla de “Jugando”.

Operación: Elegir gusano
Responsabilidad: Seleccionar un gusano.
Precondiciones: Se ha mostrado la pantalla de “Jugando”.
Postcondiciones: Se ha seleccionado un gusano.

Operación: Mostrar elección
Responsabilidad: Mostrar el gusano seleccionado.
Precondiciones: Se ha seleccionado un gusano.
Postcondiciones: Se han mostrados los posibles gusanos para elegir el segundo gusano.

Operación: Elegir segundo gusano
Responsabilidad: Seleccionar un segundo gusano entre los posibles.
Precondiciones: Se ha mostrado el gusano seleccionado y los posibles a elegir.
Postcondiciones: Se ha seleccionado un segundo gusano.

Operación: Cambio de tablero
Responsabilidad: Se cambia el tablero de juego.
Precondiciones: Se han elegido los dos gusanos.
Postcondiciones: Se ha cambiado el tablero de juego.

Capitulo 5

Diseño

5.1. Gosu

Gosu es una biblioteca para el desarrollo de videojuegos en dos dimensiones en C++ y Ruby, disponible para los sistemas Mac OS X, Microsoft Windows y GNU/Linux. La versión para C++ también está disponible para iPhone, iPod e iPad. Es una biblioteca totalmente orientada a objetos.

Esta biblioteca se ha utilizado en toda clase de juegos 2D, especialmente en competiciones cronometradas, incluyendo juegos de disparos, de scroll horizontal, juegos en perspectiva isométrica, etc. Gosu es software libre liberada bajo la licencia MIT.

Gosu ofrece interfaces sencillas para el acceso a recursos:

- Gráficos y texto 2D, con aceleración 3D por hardware (OpenGL)
- Reproducción de samples y música en diversos formatos y con transformaciones en línea.
- Soporte de dispositivos de entrada como teclado, ratón y mandos de juego. Sistema multitáctil en sistemas iOS.

A la hora de diseñar el videojuego hay que tener en cuenta todo lo que Gosu nos proporciona y lo que no. Para ello es conveniente diseñar nuestro sistema teniendo en cuenta esta herramienta:

- Lo ideal sería que Gosu incluyera una clase Tiles, en la cual cargar en memoria imágenes con movimientos (como las de la Figura 5.11). Para solucionar esto, cargaremos los distintos trozos de imágenes en una matriz, así podemos acceder rápidamente al trozo que queremos y de una manera sencilla.
- La parte gráfica de Gosu se realiza mediante coordenadas, por lo conviene incluir las coordenadas de los objetos en el diseño para luego saber donde tenemos que cargarlos.
- Gosu no incluye métodos para calcular el tiempo para las animaciones. También debemos incluirlas en el diseño.
- Gosu contiene un método para hacer la lógica del juego, otro para gestionar la carga de las imágenes en la pantalla y otro para el cálculo de los botones. Tendremos que añadir algunos atributos para después de haber seleccionado un botón hacer el calculo lógico que este nos requiere.

5.2. Diseño del sistema

Una vez especificado el comportamiento del sistema, continuaremos con el diagrama de clases del diseño y pasaremos a describir las clases que compone este sistema.

Para este análisis no he incluido aquellos métodos que no tienen importancia para el diseño del sistema, ya que sólo sirve para cálculos internos que el propio sistema no nos proporciona, como puede ser el tiempo transcurrido entre un movimiento de una animación y otro. Estos no serán incluidos para facilitar la comprensión del sistema, y para tener más claridad en el diagrama de clases.

Estos métodos y atributos no incluidos serán:

- Habrá un atributo por cada animación que nos controlará por que frame está la animación en cada momento.
- Tendremos un tiempo de inicio de animación por cada una.
- También tendremos un tiempo final para cada una.
- Tendremos los métodos observadores y modificadores de cada atributo anteriormente nombrado.
- Por último, tendremos un método que nos calculará el tiempo transcurrido desde que la animación se cambió de frame hasta el momento actual. Este nos servirá para calcular cuando ha de cambiar de frame la animación.

Los objetos que poseerán estos métodos y atributos que no hemos incluido en el diagrama de clases serán aquellos en los que se ven afectados por las animaciones:

- Uno por cada gusano del tablero.
- Los dos gusanos de la pantalla de “Selección” que nos muestra el resultado de la selección elegida.
- Las mariposas que pasan en el “Menú principal”.
- Las nubes que pasan en el “Menú principal”.
- El gusano que sale de la manzana en el “Menú principal”.

5.2.1. Diagrama de clases de diseño

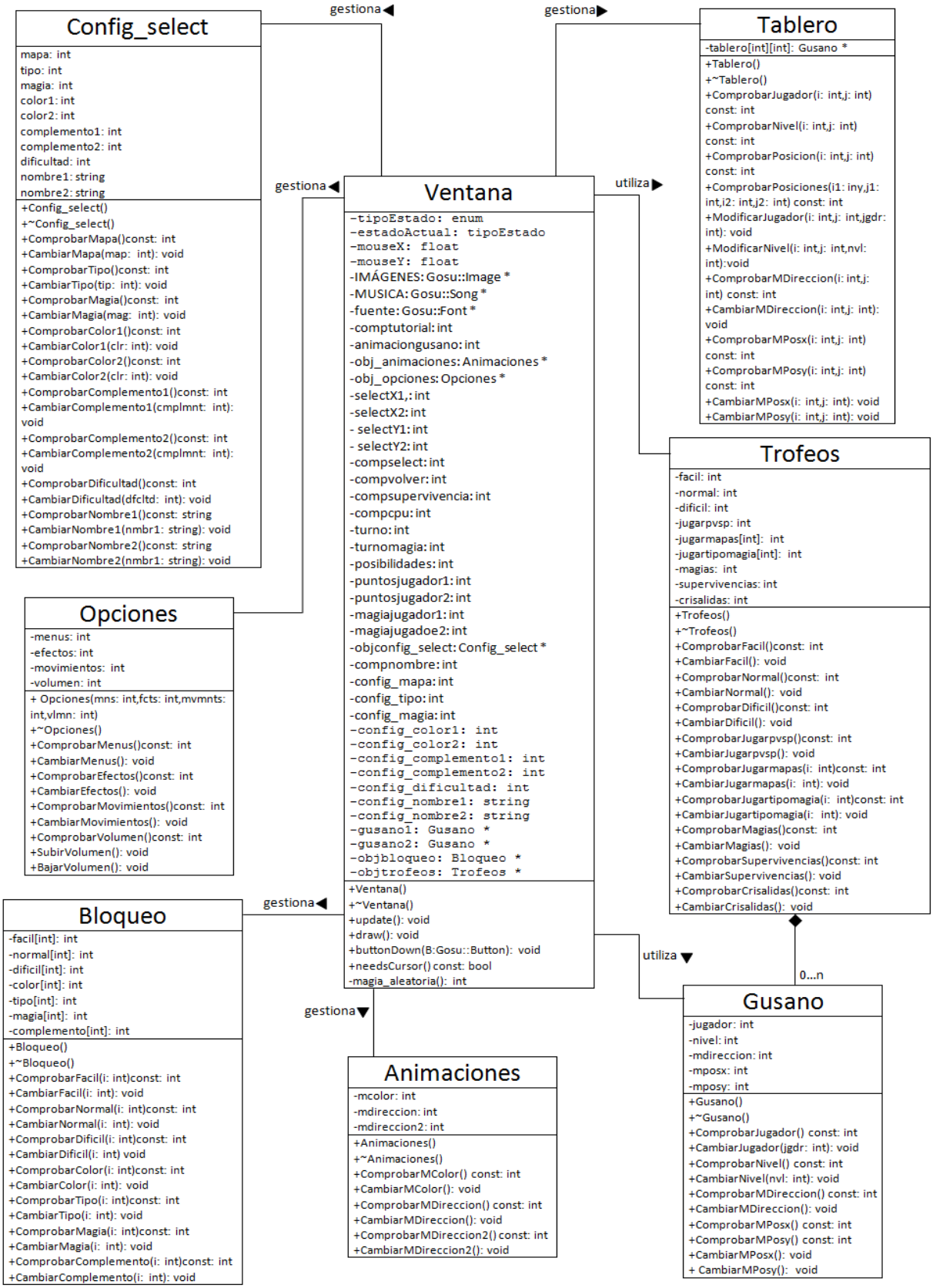

Figura 5.1: Diagrama de clases del sistema

5.2.2. Descripción de las clases

Ventana: Clase principal del juego. Gestiona el motor gráfico, el sonido, las imágenes y la lógica del juego. Es la clase predeterminada de GOSU.

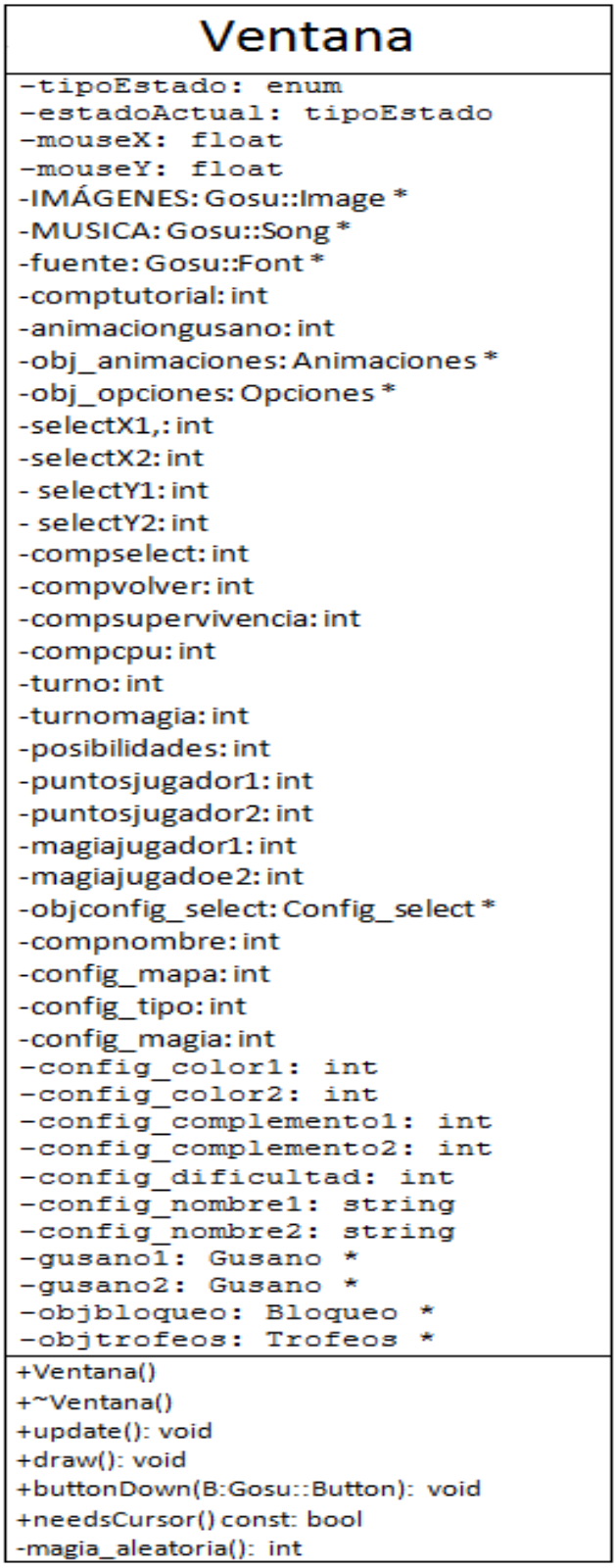

Figura 5.2: Clase Ventana

ATRIBUTOS

- **tipoEstado**: Contiene los distintos estados en los que puede estar nuestro juego.
- **estadoActual**: Contiene el estado/pantalla en el que nos encontramos.
- **mouseX**: Contiene la posición en el eje X del ratón.
- **mouseY**: Contiene la posición en el eje Y del ratón.
- **IMÁGENES**: Son punteros a imagenes del tipo GOSU. IMÁGENES representa el conjunto de imagenes necesarias para el juego.
- **MUSICA**: Son punteros a canciones del tipo GOSU. MUSICA representa el conjunto de canciones necesarias para el juego.
- **fuente**: Es un puntero a una fuente de GOSU. Es la fuente usada para el juego.
- **comptutorial**: Atributo que comprueba en que parte del tutorial estamos.
- **animaciongusano**: Guarda el tiempo que tiene que transcurrir para avanzar en la animación.
- **obj_animaciones**: Objeto de la clase Animaciones.
- **obj_opciones**: Objeto de la clase Opciones.
- **selectX1**: Guarda la coordenada X del primer gusano seleccionado del turno.
- **selectX2**: Guarda la coordenada X del segundo gusano seleccionado del turno.
- **selectY1**: Guarda la coordenada Y del primer gusano seleccionado del turno.
- **selectY2**: Guarda la coordenada Y del segundo gusano seleccionado del turno.
- **compselect**: Nos guardará en que parte de la selección nos encontramos, ya sea selección de gusano o de magia.
- **compvolver**: Atributo que comprueba si hemos pulsado en volver. Controlará que no vuelva varias pantallas a la vez.
- **compsupervivencia**: Comprueba si ha realizado el movimiento llamado “Supervivencia”.
- **compcpu**: Comprueba si estamos en la selección para jugar contra la CPU.
- **turno**: Guardará a que jugador le pertenece el turno.
- **turnomagia**: Guardará cuantos turnos queda para que se asigne una nueva magia.
- **posibilidades**: Guarda las posibilidades de movimientos que hay en el tablero.
- **puntosjugador1**: Es el marcador del jugador 1.
- **puntosjugador2**: Es el marcador del jugador 2.
- **magiajugador1**: Guarda que magia contiene el jugador 1.
- **magiajugadoe2:** Guarda que magia contiene el jugaro 2.
- **objconfig_select**: Objeto de la clase Config_select.
- **compnombre**: comprueba que el nombre no esté repetido.
- **config_mapa**: Guarda que mapa está seleccionado.
- **config_tipo**: Guarda que tipo de juego está seleccionado.
- **config_magia**: Guarda que frecuencia de magias está seleccionada.
- **config_color1**: Guarda el color para el jugador 1 seleccionado.
- **config_color2**: Guarda el color para el jugador 2 seleccionado.
- **config_complemento1**: Guarda el complemento para el jugador 1 seleccionado.
- **config_complemento2**: Guarda el complemento para el jugador 2 seleccionado.
- **config_dificultad**: Guarda la dificultad seleccionada para jugar contra CPU.
- **config_nombre1**: Guarda el nombre del jugador 1.
- **config_nombre2**: Guarda el nombre del jugador 2.
- **gusano1**: Muestra lo seleccionado por el jugador 1.
- **gusano2**: Muestra lo seleccionado por el jugador 2.
- **objbloqueo**: Objeto de la clase Bloqueo.
- **objtrofeos**: Objeto de la clase Trofeos.

MÉTODOS

- **Ventana()**: Constructor de la clase Ventana. Inicializa todos los atributos y guarda espacio en memoria para los objetos, las imagenes, etc...
- **~Ventana()**: Destructor de la clase Ventana. Libera la memoria.
- **update()**: Se encarga de controlar los cálculos y la lógica del juego.
- **draw()**: Se encarga de mostrar las imagenes por pantalla.
- **buttonDown(B:Gosu::Button)**: Se encarga de la gestión de los botones del juego.
- **needsCursor() const**: Nos dice si el cursor está o no visible.
- **magia_aleatoria()**: Genera una magia aleatoria para asignársela a un jugador.

Tablero: Clase que gestiona el tablero de juego. Esta clase estará compuesta por una matriz, y contendrá los gusanos con los que jugaremos.

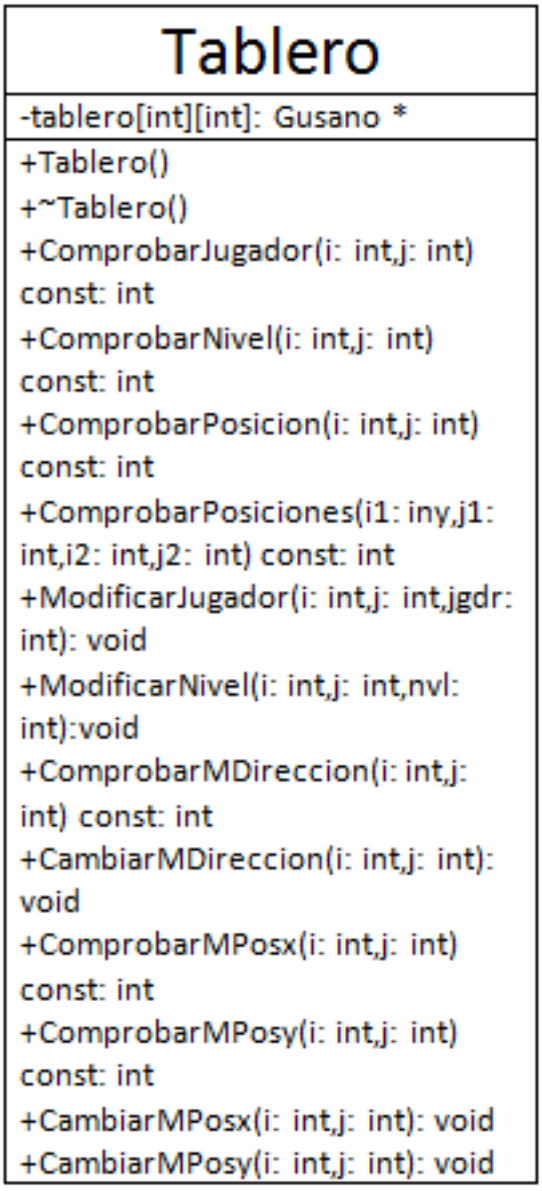
Tablero
-tablero[int][int]: Gusano *
+Tablero()
+~Tablero()
+ComprobarJugador(i: int,j: int) const: int
+ComprobarNivel(i: int,j: int) const: int
+ComprobarPosicion(i: int,j: int) const: int
+ComprobarPosiciones(i1: iny,j1: int,i2: int,j2: int) const: int
+ModificarJugador(i: int,j: int,jgdr: int): void
+ModificarNivel(i: int,j: int,nvl: int):void
+ComprobarMDireccion(i: int,j: int) const: int
+CambiarMDireccion(i: int,j: int): void
+ComprobarMPosx(i: int,j: int) const: int
+ComprobarMPosy(i: int,j: int) const: int
+CambiarMPosx(i: int,j: int): void
+CambiarMPosy(i: int,j: int): void

Figura 5.3: Clase Tablero

ATRIBUTOS

- **tablero[int][int]**: Matriz que contiene punteros a objetos Gusano.

MÉTODOS

- **Tablero()**: Constructor de la clase Tablero.
- **~Tablero()**: Destructor de la clase Tablero.
- **ComprobarJugador(i: int,j: int) const**: Comprueba que jugador hay en la posición (i,j).
- **ComprobarNivel(i: int,j: int) const**: Comprueba que nivel tiene el jugador hay en la posición (i,j).
- **ComprobarPosicion(i: int,j: int) const**: Comprueba que el gusano que hay en la posición (i,j) sea posible de elegir.
- **ComprobarPosiciones(i1: iny,j1: int,i2: int,j2: int) const**: Comprueba que entre esos dos gusanos hay un movimiento posible.
- **ModificarJugador(i: int,j: int,jgdr: int)**: Modifica el jugador del gusano de la posición (i,j).
- **ModificarNivel(i: int,j: int,nvl: int)**: Modifica el nivel del gusano de la posición (i,j).
- **ComprobarMDireccion(i: int,j: int) const**: comprueba la dirección de la mariposa.
- **CambiarMDireccion(i: int,j: int)**: Cambia la dirección de la mariposa.
- **ComprobarMPosx(i: int,j: int) const**: Comprueba la posición X de la mariposa.
- **ComprobarMPosy(i: int,j: int) const**: Comprueba la posición Y de la mariposa.
- **CambiarMPosx(i: int,j: int)**: Cambia la posición X de la mariposa.
- **CambiarMPosy(i: int,j: int)**: Cambia la posición Y de la mariposa.

Gusano: Clase que contendrá todas las características y el comportamiento de los distintos gusanos que encontraremos durante el juego. Estos estarán colocados en el tablero anteriormente mencionado. También aparecerán en la pantalla de "Selección" para mostrarnos cuál es el resultado de las opciones escogidas para nuestros gusanos.

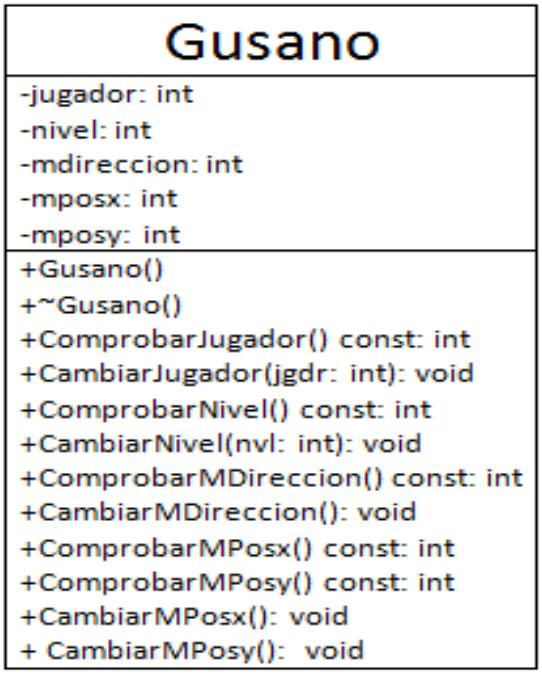

Figura 5.4: Clase Gusano

ATRIBUTOS

- **jugador**: Guardará a que jugador pertenece este gusano.
- **nivel**: Guardará el nivel del gusano.
- **mdireccion**: Contiene la dirección de la mariposa.
- **mposx**: Contiene la posición X de la mariposa.
- **mposy**: Contiene la posición Y de la mariposa.

MÉTODOS

- **Gusano()**: Constructor de la clase Gusano.
- **~Gusano()**: Destructor de la clase Gusano.
- **ComprobarJugador() const**: Comprueba que jugador es.
- **ComprobarNivel() const**: Comprueba que nivel tiene el gusano.
- **ModificarJugador()**: Modifica el jugador del gusano.
- **ModificarNivel()**: Modifica el nivel del gusano.
- **ComprobarMDireccion() const:** comprueba la dirección de la mariposa.
- **CambiarMDireccion()**: Cambia la dirección de la mariposa.
- **ComprobarMPosx() const**: Comprueba la posición X de la mariposa.
- **ComprobarMPosy() const**: Comprueba la posición Y de la mariposa.
- **CambiarMPosx()**: Cambia la posición X de la mariposa.
- **CambiarMPosy()**: Cambia la posición Y de la mariposa.

Animaciones: Clase que se encargará de gestionar el tiempo en las animaciones y en los frames en los que se encuentran estas.

Figura 5.5: Clase Animaciones

ATRIBUTOS

- **mcolor**: Guarda el color de la mariposa que sale en el "Menú principal".
- **mdireccion**: Guarda la dirección (derecha, izquierda) a la que va la mariposa en el "Menú principal".

- **mdireccion2**: Guarda la dirección (arriba, abajo) a la que va la mariposa en el "Menú principal".

MÉTODOS

- **Animaciones()**: Constructor de la clase Animaciones.
- **~Animaciones()**: Destructor de la clase Animaciones.
- **ComprobarMColor() const**: Comprueba el color de la mariposa en el "Menú principal".
- **CambiarMColor()**: Cambia el color de la mariposa en el "Menú principal".
- **ComprobarMDireccion() const**: Comprueba la dirección (derecha, izquierda) de la mariposa en el "Menú principal".
- **CambiarMDireccion()**: Cambia la dirección (derecha, izquierda) de la mariposa en el "Menú principal".
- **ComprobarMDireccion2() const**: Comprueba la dirección (arriba, abajo) de la mariposa en el "Menú principal".
- **CambiarMDireccion2()**: Cambia la dirección (arriba, abajo) de la mariposa en el "Menú principal".

Bloqueo: Clase que gestionará los distintos componentes del juego que hemos desbloqueado a medida que nos lo hemos pasado. Guardará en un fichero los avances que hemos logrado.

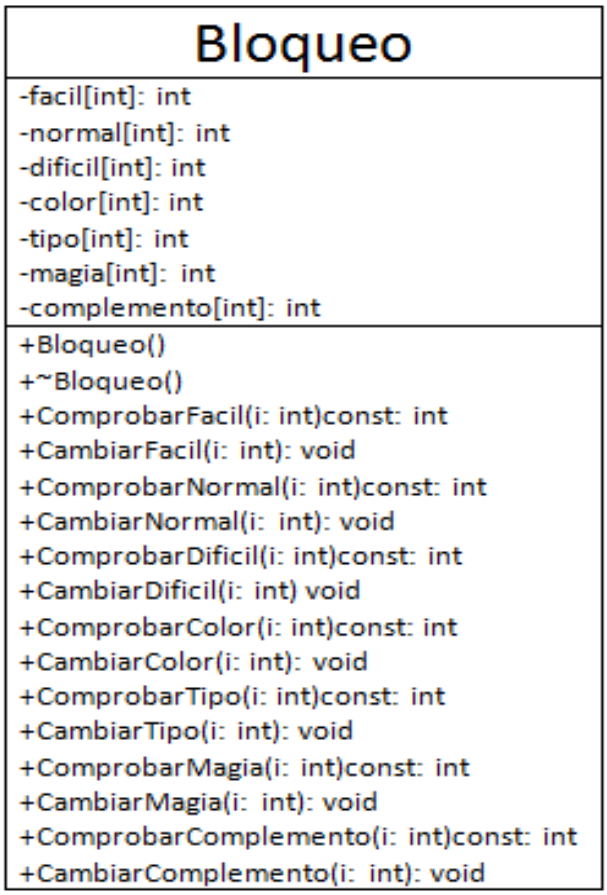

Figura 5.6: Clase Bloqueo

ATRIBUTOS

- **facil[int]**: Guarda si se ha completado las pantallas contra la CPU en fácil.
- **normal[int]**: Guarda si se ha completado las pantallas contra la CPU en normal.
- **dificil[int]**: Guarda si se ha completado las pantallas contra la CPU en dificil.
- **color[int]**: Guarda los colores desbloqueados.
- **tipo[int]**: Guarda los tipos de juegos desbloqueados.
- **magia[int]**: Guarda las frecuencia de magias desbloqueadas.
- **complemento[int]**: Guarda los complementos desbloqueados.

MÉTODOS

- **Bloqueo()**: Constructor de la clase Bloqueo.
- **~Bloqueo()**: Destructor de la clase Bloqueo.
- **ComprobarFacil(i: int)const**: Comprobar si se ha completado las pantallas contra la CPU en fácil.
- **CambiarFacil(i: int)**: Cambiar si se ha completado las pantallas contra la CPU en fácil.
- **ComprobarNormal(i: int) const**: Comprobar si se ha completado las pantallas contra la CPU en normal.
- **CambiarNormal(i: int)**: Cambiar si se ha completado las pantallas contra la CPU en normal.
- **ComprobarDificil(i: int) const**: Comprobar si se ha completado las pantallas contra la CPU en dificil.
- **CambiarDificil(i: int)**: Cambiar si se ha completado las pantallas contra la CPU en dificil.
- **ComprobarColor(i: int) const**: Comprueba si el color se ha desbloqueado.
- **CambiarColor(i: int)**: Cambiar si el color se ha desbloqueado.
- **ComprobarTipo(i: int) const**: Comprobar si el tipo de juego se ha desbloqueado.
- **CambiarTipo(i: int)**: Cambiar a desbloqueado el tipo de juego.
- **ComprobarMagia(i: int) const**: Comprueba si la frecuencia de magia ha sido desbloqueada.
- **CambiarMagia(i: int)**: Pone la frecuencia de magia desbloqueada.
- **ComprobarComplemento(i: int) const**: Comprueba si el complemento fué desbloqueado.
- **CambiarComplemento(i: int)**: Pone el complemento como desbloqueado.

Config_select: Clase que se encarga de memorizar la última selección realizada en la pantalla de "Selección". Guardará la información en un fichero.

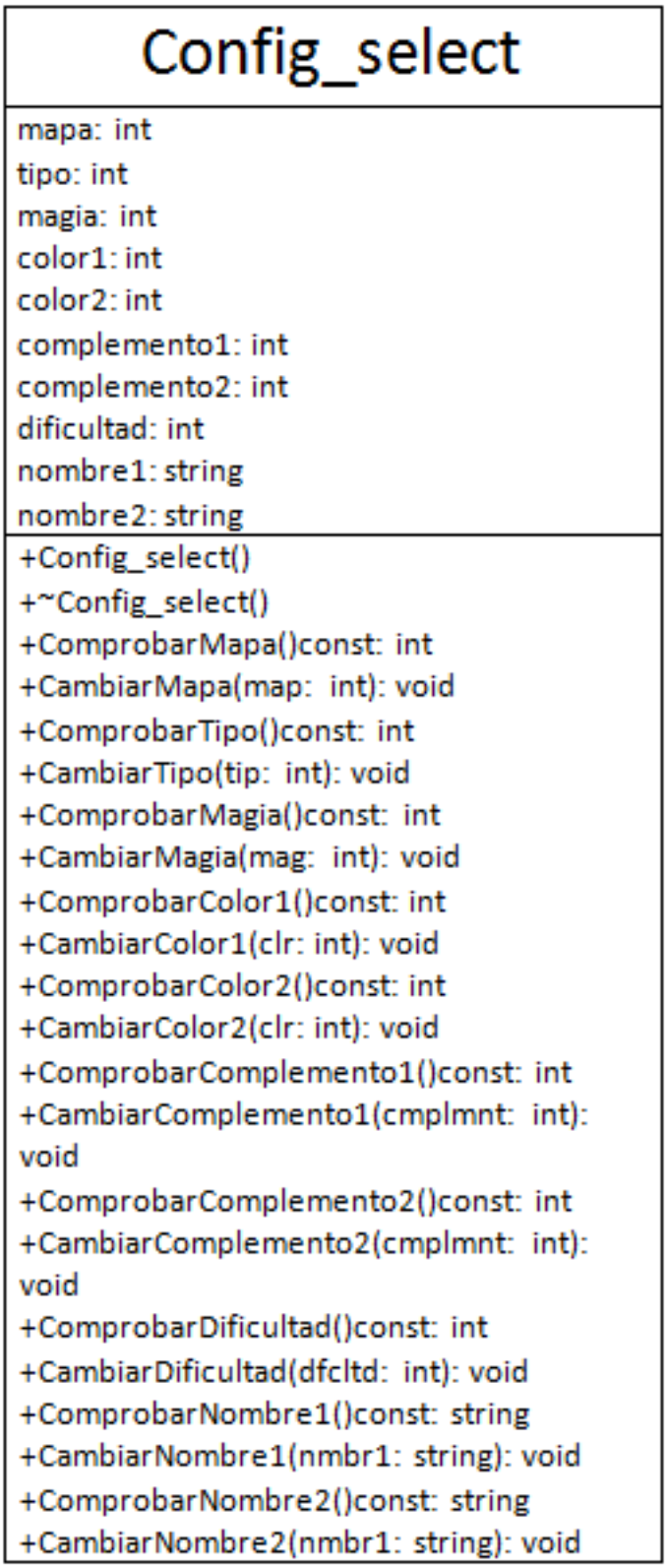

Figura 5.7: Clase Config_select

ATRIBUTOS

- **mapa:** Contiene el mapa seleccionado.
- **tipo:** Contiene el tipo de juego seleccionado.
- **magia**: Contiene la frecuencua de magia seleccionada.
- **color1**: Contiene el color del jugador 1 seleccionado.
- **color2**: Contiene el color del jugador 2 seleccionado.
- **complemento1**: Contiene el complemento del jugador 1 seleccionado.
- **complemento2:** Contiene el complemento del jugador 2 seleccionado.
- **dificultad:** Contiene la dificultad escogida para jugar contra la CPU.
- **nombre1**: Contiene el nombre del jugador 1.
- **nombre2:** Contiene el nombre del jugador 2.

MÉTODOS

- **Config_select()**: Constructor de la clase Config_select.
- **~Config_select()**: Destructor de la clase Config_select.
- **ComprobarMapa() const**: Comprobar el mapa seleccinado.
- **CambiarMapa(map: int)**: Cambiar de mapa.
- **ComprobarTipo() const**: Comprobar el tipo de juego seleccionado.
- **CambiarTipo(tip: int)**: Cambiar de tipo de juego.
- **ComprobarMagia() const**: Comprueba la frecuencia de magia seleccionada.
- **CambiarMagia(mag: int)**: Cambia la frecuencia de magia.
- **ComprobarColor1() const**: Comprueba el color del jugador 1.
- **CambiarColor1(clr: int)**: Cambia el color del jugador 1.
- **ComprobarColor2() const**: Comprueba el color del jugador 2.
- **CambiarColor2(clr: int)**: Cambia el color del jugador 2.
- **ComprobarComplemento1() const**: Comprueba el complemento del jugador 1.
- **CambiarComplemento1(cmplmnt: int)**: Cambia el complemento del jugador 1.
- **ComprobarComplemento2() const**: Comprueba el complemento del jugador 2.
- **CambiarComplemento2(cmplmnt: int)**: Cambia el complemento del jugador 2.
- **ComprobarDificultad() const**: Comprueba la dificultad.
- **CambiarDificultad(dfcltd: int)**: Cambia la dificultad.
- **ComprobarNombre1() const**: Comprueba el nombre del jugador 1.
- **CambiarNombre1(nmbr1: string)**: Cambia el nombre del jugador 1.
- **ComprobarNombre2() const:** Comprueba el nombre del jugador 2.
- **CambiarNombre2(nmbr1: string)**: Cambia el nombre del jugador 2.

Opciones: Clase que se encargará de guardar en un fichero la última configuración seleccionada en la pantalla de “Opciones”.

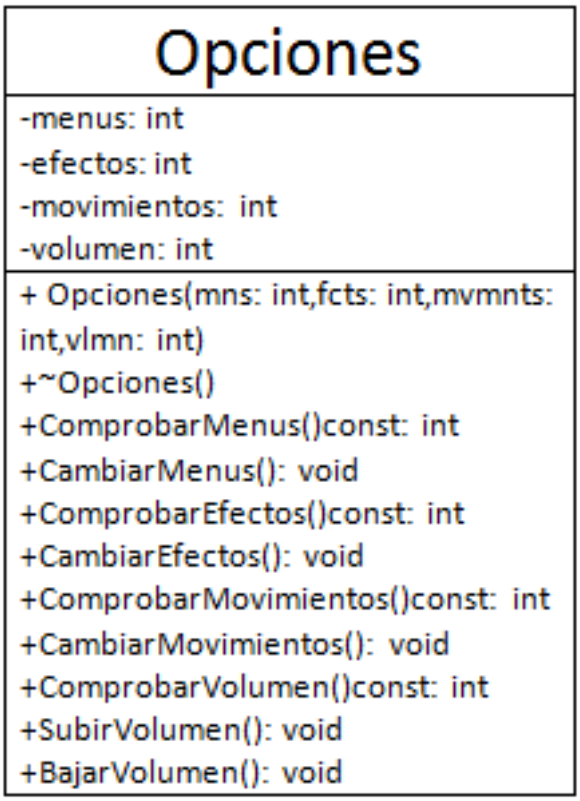

Figura 5.8: Clase Opciones

ATRIBUTOS

- **menus**: Guarda si las animaciones de los menús están o no activas.
- **efectos:** Guarda si las animaciones de los efectos están o no activas.
- **movimientos**: Guarda si las animaciones de los movimientos están o no activas.
- **volumen:** Guarda el nivel del volumen de la música.

MÉTODOS

- **Opciones(mns: int,fcts: int,mvmnts: int,vlmn: int):** Costructor de la clase Opciones.
- **~Opciones():** Destructor de la clase Opciones.
- **ComprobarMenus() const**: Comprueba si las animaciones de menús están activas o no.
- **CambiarMenus()**: Cambia si las animaciones de menús están activas o no.
- **ComprobarEfectos() const**: Comprueba si las animaciones de efectos están activas o no.
- **CambiarEfectos()**:Cambia si las animaciones de efectos están activas o no.
- **ComprobarMovimientos() const**: Comprueba si las animaciones de movimientos están activas o no.
- **CambiarMovimientos()**: Cambia si las animaciones de movimientos están activas o no.
- **ComprobarVolumen() const**: Comprueba el nivel del volumen.
- **SubirVolumen()**: Sube el nivel del volumen.
- **BajarVolumen()**: Baja el nivel del volumen.

Trofeos: Clase que se encargará de guardar en un fichero los distintos retos que hemos alcanzado jugando, que se nos mostrará en forma de trofeos en la pantalla "Trofeos".

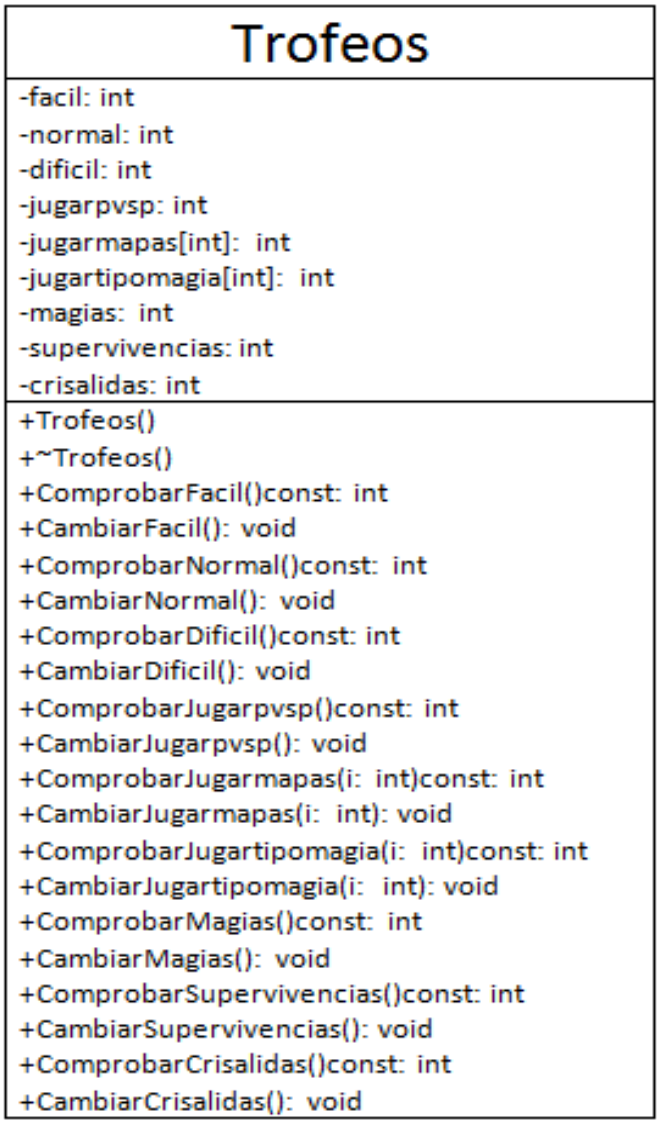

Figura 5.9: Clase Trofeos

ATRIBUTOS

- **facil**: Guarda por qué mapa va en dificultad fácil.
- **normal:** Guarda por qué mapa va en dificultad normal.
- **dificil**: Guarda por qué mapa va en dificultad dificultad.
- **jugarpvsp**: Guarda cuantas veces se ha jugado en Player versus Player.
- **jugarmapas[int]**: Guarda en que mapas se ha jugado en Player versus Player..
- **jugartipomagia[int]**: Guarda que tipo de juego o frecuencias de magia se han jugado en Player versus Player..
- **magias:** Guarda cuantas magias se han utilizado.
- **supervivencias:** Guarda cuantos movimientos de supervivencias se han realizado.
- **crisalidas**: Guarda cuantos movimientos de crisálidas se han realizado.

MÉTODOS

- **Trofeos()**: Constructor de la clase Trofeos.

- **~Trofeos()**: Destructor de la clase Trofeos.
- **ComprobarFacil() const**: Comprueba que fase en fácil está desbloqueada.
- **CambiarFacil()**: Cambia el bloqueo en fácil.
- **ComprobarNormal() const**: Comprueba que fase en normal está desbloqueada.
- **CambiarNormal()**: Cambia el bloqueo en normal.
- **ComprobarDificil() const**: Comprueba que fase en dificil está desbloqueada.
- **CambiarDificil()**: Cambia el bloqueo en dificil.
- **ComprobarJugarpvsp() const**: Comprueba cuantas veces se jugó en Player versus Player.
- **CambiarJugarpvsp()**: Cambia las veces jugadas.
- **ComprobarJugarmapas(i: int) const**: Comprueba qué mapas se jugaron en Player versus Player.
- **CambiarJugarmapas(i: int)**: Cambia el mapa jugado.
- **ComprobarJugartipomagia(i: int) const**: Comprueba que tipo de juego o frecuencia de magias se ha usado en Player versus Player.
- **CambiarJugartipomagia(i: int)**: Cambia lo usado.
- **ComprobarMagias() const**: Comprueba el número de magias usadas.
- **CambiarMagias()**: Suma 1 a las magias usadas.
- **ComprobarSupervivencias() const:** Comprueba el número de supervivencias realizadas.
- **CambiarSupervivencias()**: Suma uno a las supervivencias realizadas.
- **ComprobarCrisalidas() const**: Comprueba el número de crisálidas realizadas.
- **CambiarCrisalidas()**: Suma uno a las crisálidas realizadas.

5.3 Diseño multimedia

A continuación mostraré y contaré algunas de las desiciones sobre el diseño multimedio o diseño gráfico.

5.3.1. Personajes

Para crear una gran variedad de elección decidí crear 5 colores distintos para los gusanos, y 15 complementos para ponerles.

Figura 5.10: Personajes

Para realizar estos pernajes usé la herramienta GraphicsGale [4]. Luego los anime usando TiledMap [3]. El resultado de una de las animaciones es el siguiente:

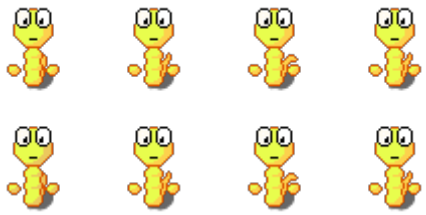

Figura 5.11: Personajes animados

Cuando estos personajes obtienes 5 manzanas se transforman en crisálidas:

Figura 5.12: Crisalida

Tras terminar el juego estos gusanos se tranformarán en mariposas. Aquí tenemos un ejemplo de estas mariposas:

Figura 5.13: Mariposas animadas

5.3.2. Mapas

Podremos elegir en el juego entre 12 mapas. Cada mapa supondrá un reto distinto a superar, ya que la diferencia entre los gusanos que poseemos, el terreno, y la conexión entre ellos hará una gran variedad. A esto se le suma que hay 5 tipos de juego distinto y 5 frecuencias de magias a elegir. Esto hace del juego tener una gran variedad.

Tendremos que elegir entre un bosque, un gran puente, un pequeño lago, unas islas, una muralla, el monte, unos castillos, un desierto, el interior de un volcán, una pradera nevada, un pantano o una llanura llena de flores.

Estos mapas fueron realizado con los mismos programas anteriormente nombrados.

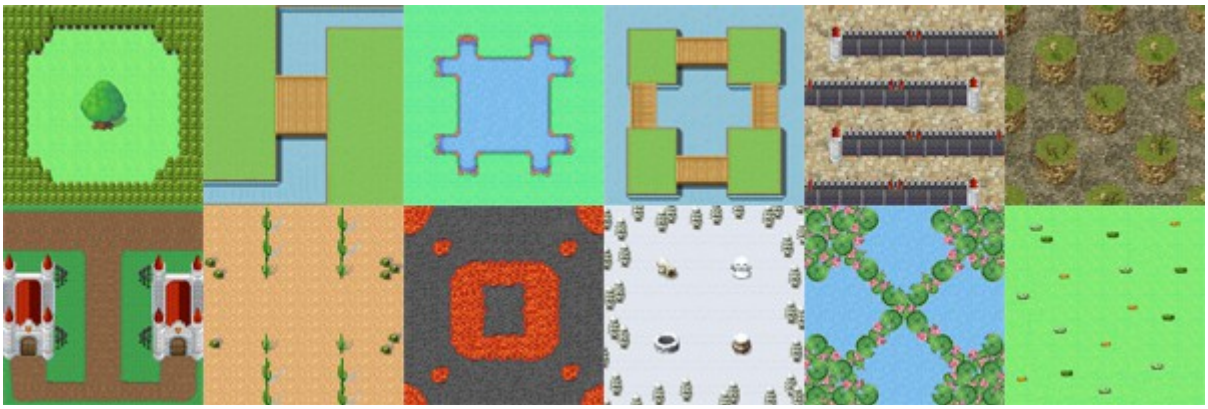

Figura 5.14: Mapas

5.3.3. Música

Para la música no hemos utilizado ningun programa. La música ha sido obtenida de una página web [1] donde se comporte música y sonidos con licencia libre.

Algunos de los autores seleccionados para poner música de ellos en mi juego son:

- Schwartz Germán Andrés
- Pep Nuñez
- Roch Sergio Adrián
- Primitiu Safont Richarte
- Petrus

5.3.4. Menús

Para realizar los menús también usé los mismo programas. En este caso, he contado con la colaboración de Francisco Javier Rodriguez Romero, el cual ha desarrollado algunas imagenes para mí utilizando GIMP.

Capitulo 6

Implementación

Como ya he comentado con anterioridad, este proyecto está implementado con C++ y GOSU. Me decidí por este lenguaje y esta librería puesto que ya la conocía de la universidad, y este proyecto me permitiría conocerla aún más a fondo.

La lógica del juego y la implementación de los menús viene dado por nuestro análisis del sistema, así que a continuación comentaremos algunos detalles de la programación de este, como pueden ser la Inteligencia artificial, como acceder al tablero de juego, un diagrama de estados del juego y por último las colisiones.

6.1. Inteligencia Artificial

La inteligencia artificial es algo muy importante en un juego que tenga un modo de juego contra el propio ordenador. Al jugar contra la CPU, el jugador debe verse retado a superar distintos obstáculos. Para ello he creado tres niveles de inteligencia artificial, el cual irá perfeccionando los movimientos y añadiendo otros nuevos.

Toda la programación de estas inteligencias artificiales podremos verlas en el anexo C.

6.1.1. Fácil

Esta dificultad esta pensada para aquellos jugadores que estén comenzando a jugar, y necesitan cogerle el truco al juego y experimentar contra una dificultad más fácil.

Los movimientos que realiza esta dificultad son menos completos. Se basa en mirar si hay algún movimiento para realizar una crisálida con un gusano de 4 manzanas, y en el caso de no haberlo realizará un movimiento escogiendo un gusano al azar. Si puede moverá con puntos, y si no lo hará sin realizar nungún punto.

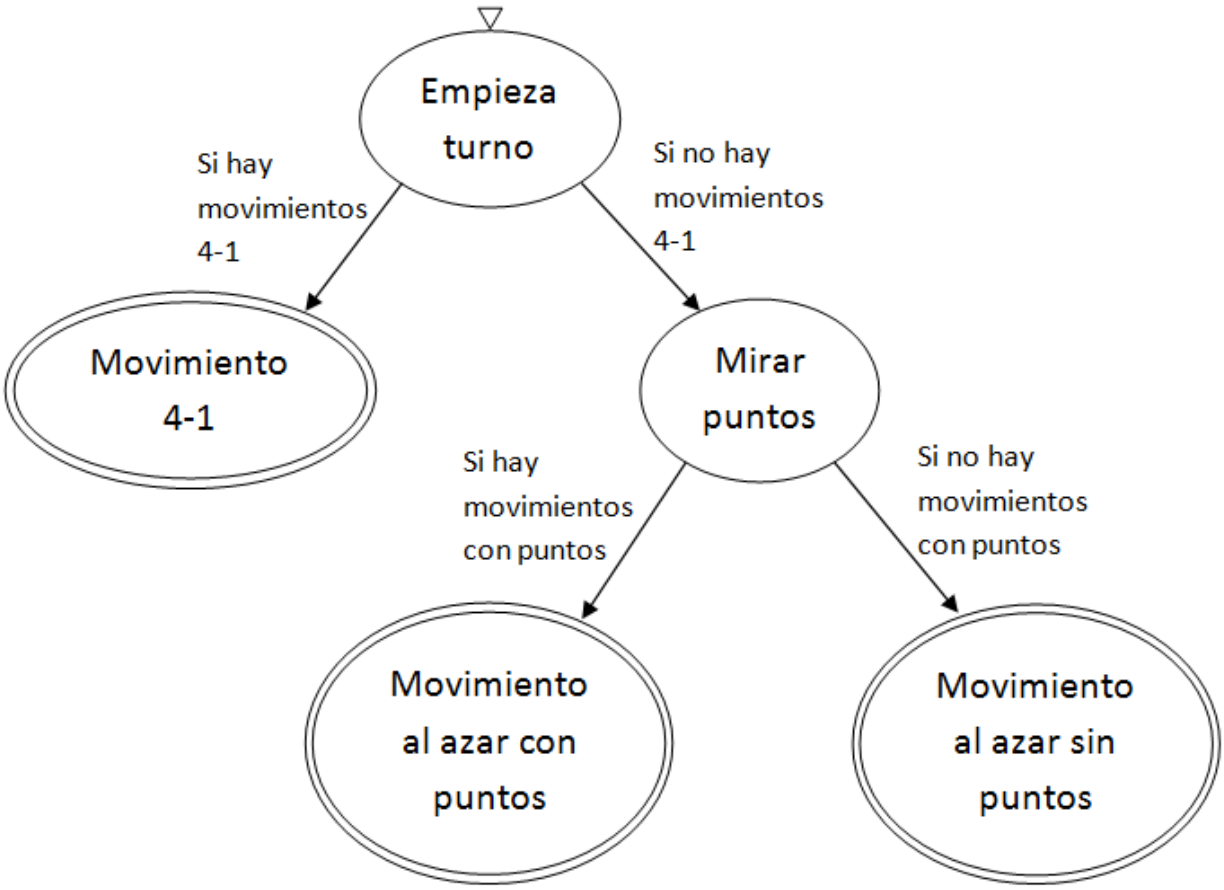
Empieza turno
Si hay movimientos 4-1
Si no hay movimientos 4-1
Movimiento 4-1
Mirar puntos
Si hay movimientos con puntos
Si no hay movimientos con puntos
Movimiento al azar con puntos
Movimiento al azar sin puntos

Figura 6.1: IA Fácil

6.1.2. Normal

Esta dificultad se ve con movimientos más completos. Estos siempre son realizados comprobando si se puede conseguir puntos a su favor.

La única mejora que tiene esta categoría no es realizar movimientos más eficientes. También añade un nuevo movimiento, comprobar si se puede realizar una crisálida, pero en vez de con un gusano de cuatro manzanas y uno de una, con un gusano de dos manzanas y otro de tres manzanas. Este movimiento lo realizará a su favor.

Otro movimiento que se ha añadido en esta dificultad es realizar magias isntantáneas. Como antes hemos dicho, habrá dos tipos de magias: Las instantáneas y las que no lo són. Las primeras se realizará de forma automática al activar la magia. Las segundas una vez activa necesitará de la elección de un gusano, lo que puede dar lugar a una amplia gama de estrategias.

En el caso de jugar contra la inteligencia artificial no sabremos cual es la magia que esta posee. La usará de manera eficiente y sin realizar jugadas a largo plazo tal como prodría hacer un jugador. En el caso de jugar contra otro jugador, la magia será publica a los dos jugadores todo el tiempo, por lo que los dos contrincantes podrán pensar sus estrategias de antemano.

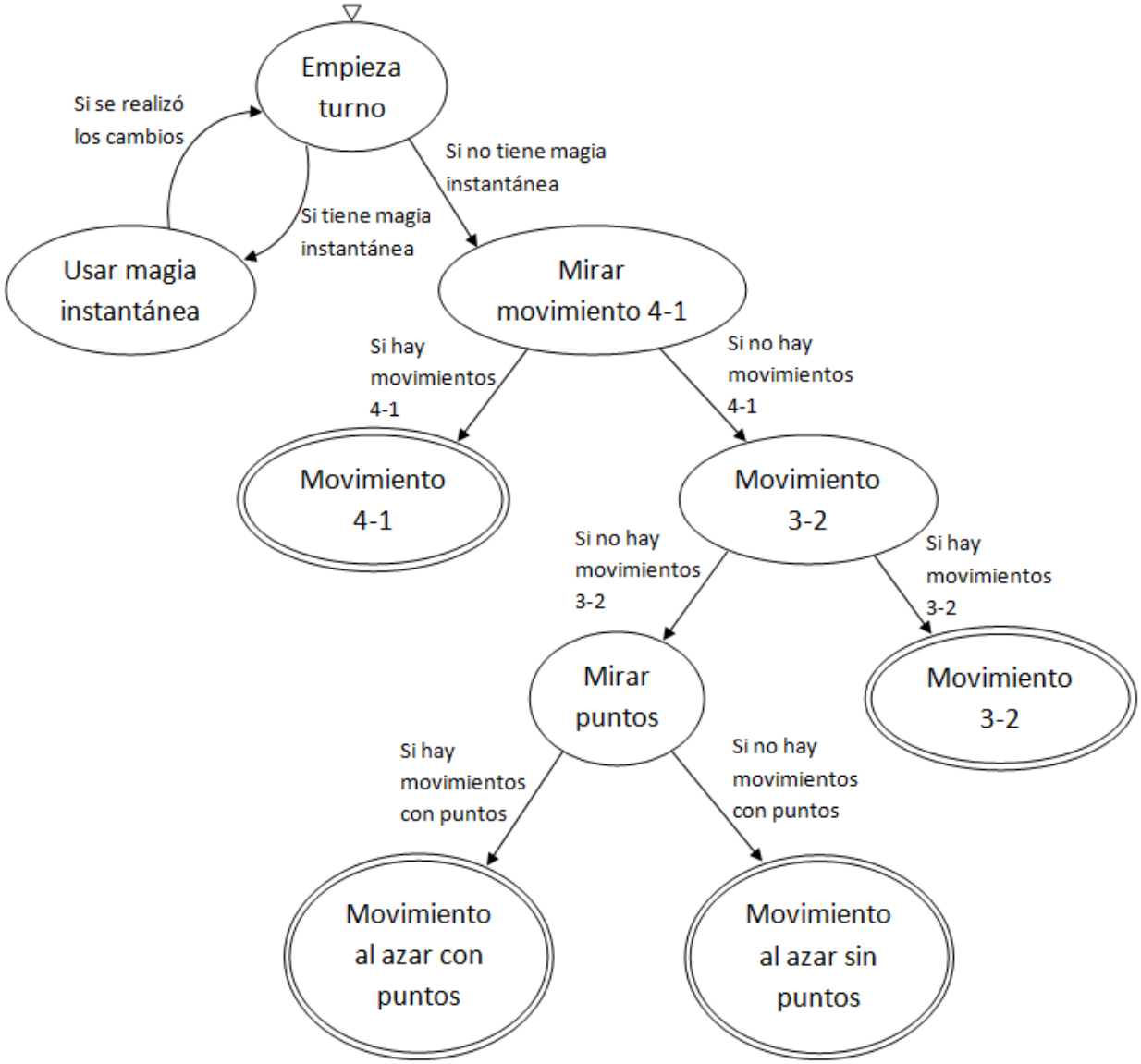

Figura 6.2: IA Normal

6.1.3. Dificil

Es la dificultad más completa de todas. A parte de realizar los movimientos anteriores tiene una serie de movimientos nuevos.

También usará las magias que no son instantáneas. Siempre las usará buscando la forma de hacer puntos. Otro movimiento es poner un gusano de su tipo de dos manzanas al lado de uno de tres, para hacer luego el movimiento que consigue una crisálida.

Otro movimiento que realizará será separar los gusanos de dos manzanas y de tres manzanas del jugador. Realizará una jugada con una probabilidad del 10% que será ponerle una trampa al jugador. Le pondrá un gusano de 2 manzanas a otro de dos manzanas, para que el jugador cometa el error de poner a uno de esos gusano a 3 manzanas y luego realizar un movimiento que realize una crisálida.

Figura 6.3: IA Dificil

6.2. Colisiones

Las colisiones se producen cuando un objeto del juego entra en contacto con otro. Son parte importante de la mayoria de los juegos que existen hoy en día.

Este juego no contiene colisiones durante la partida, pero una vez finalizada esta, aquellos gusanos que han sido transformados en crisálidas saldrán de estas y volarán libremente por la pantalla. Dado que la lógica de este juego se programa por coordenadas, habría que controlar que las mariposas no se salieran de la pantalla de juego. Para ello se establece unas colisiones con los bordes de la pantalla. Al llegar la mariposa a uno de los bordes, este cambiará el sentido y la dirección en la que volaba. Se podría ver como varias mariposas volando dentro de una caja del tamaño de la pantalla de juego.

6.3. Tablero

A la hora de implementar el juego, uno de los problemas con el que me encontré fué como saber cual es el gusano que estabamos seleccionando con el ratón.

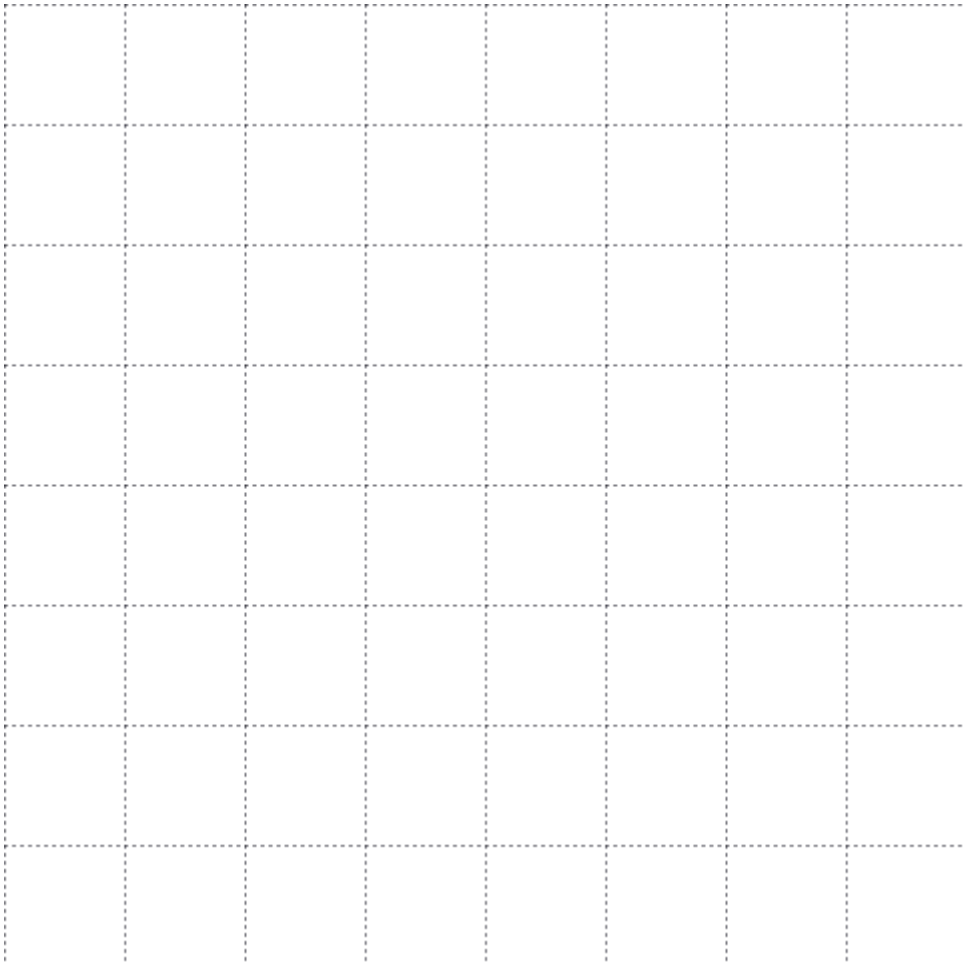

Figura 6.4: Tablero

El tablero de juego está formado por una matriz de 8 filas y 8 columnas como podemos ver en la figura 6.4.

La dimensión de cada casilla del tablero es de 60x60 pixeles. Sabiendo esto, para acceder a una de las casillas tendremos que calcular el módulo de las coordenadas con 60:

i= coordi mod 60
j= coordj mod 60

Para saber la localización de un gusano tendríamos que hacer lo contrario:

coordi = i*60
coordj= j*60

Con esto se solucionaría nuestro problema para saber acceder a un gusano en el tablero o para saber donde está un gusano del tablero.

6.4. Diagrama de estados de juego

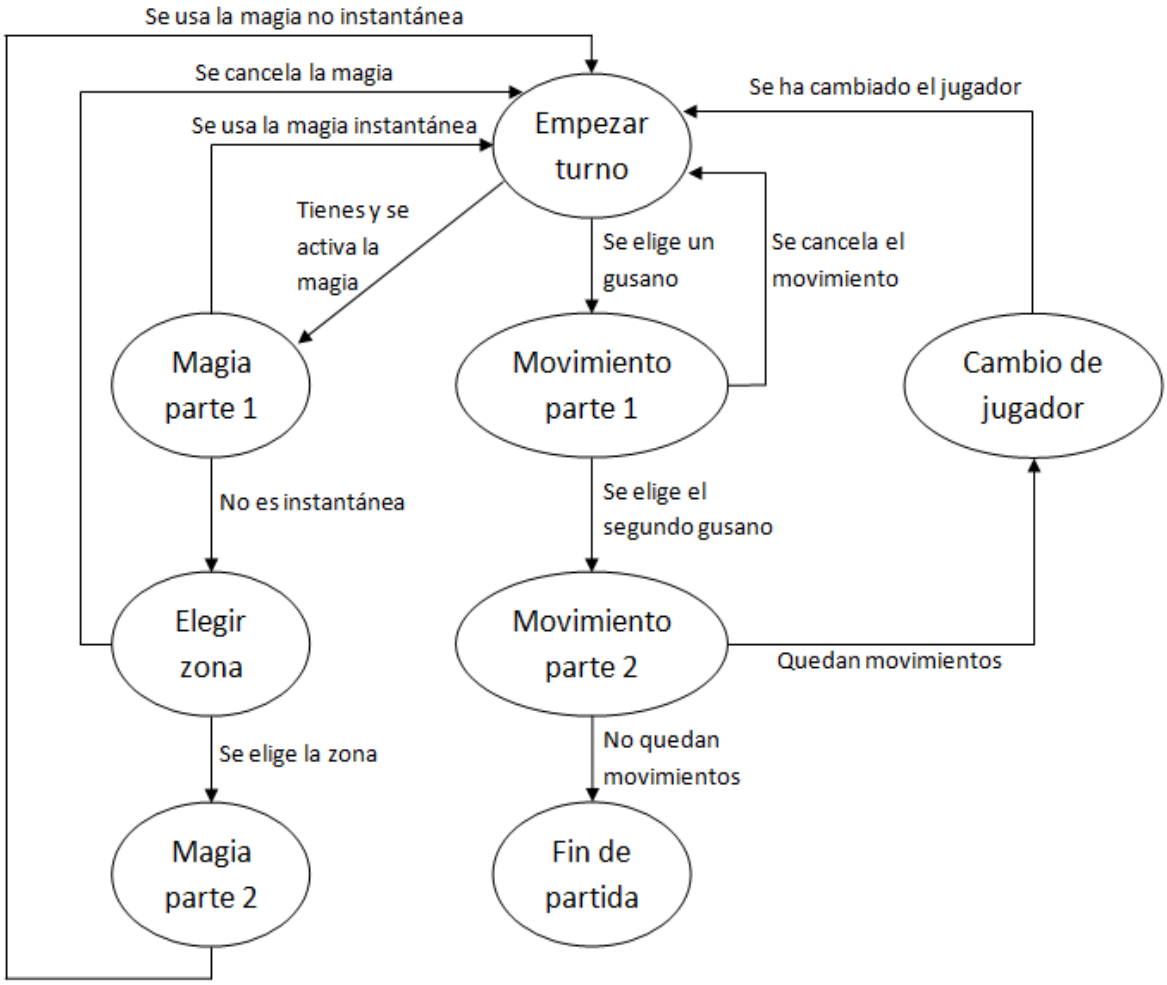

Figura 6.5: Diagrama de estados de juego

Este diagrama nos muestra como sería una partida en el juego. Uno de los jugadores empezaría su turno. Este puede elegir entre dos opciones. Usar una magia (si tiene) y elegir los dos gusanos que forman un movimiento, o sólo elegir los gusanos. Una ves realizado su movimiento, si aún queda alguna posibilidad de movimiento, se cambiará el jugador y empezará su turno, sino se terminará la partida.

Capitulo 7

Pruebas

Las pruebas son parte importante del desarrollo de cualquier software. Estas suelen realizarse cada vez que se completa algunos de los objetivos para verificar que todo va funcionando correctamente. En cambio, yo he decidido realizar sólo dos tipos de pruebas:

7.1. Pruebas incrementales

Estas pruebas las he ido realizando durante todo el desarrollo del software. Han consistido en probar cada parte de este software por separado, sin esperar a tener un bloque determinado de código para probar. Asi he realizado pruebas desde los componentes más básicos (pruebas unitarias), hasta componentes mas complejos. Una vez que un componente funcionaba correctamente incluía este en el código del juego. Una vez implantado el nuevo componente en el juego (ya fuera mas simple o complejo) realizaba otra prueba para comprobar que este componente funcionaba correctamente junto al resto del sistema.

De esta forma he ido componiendo un sistema que cada vez era más complejo, pero asegurandome que los diferentes componentes funcionanban correctamente:

- Lo primero que desarrolle fueron los menus. Para ello fuí programando las distintas pantallas del menú a la vez que hacía la parte del diseño gráfico. Una vez terminado comprobé que todo funcionaba bien y entonces se me ocurrió una idea: hacer los menús más dinamicos y interactivo. En la pantalla principal puse una mariposa que vuela por la pantalla y que al pulsar con el ratón sobre ella cambie su dirección. También agregé nubes moviendose por el fondo. En la pantalla de selección agregé que el gusano estuviera mirando siempre al puntero del ratón, y que al entrar en sus ojos los cerrara. Y por último, en el menú de configurar la partida introducí que el gusano que muestra tu selección fuera animado. Una vez terminado todos estos cambios comprobé una vez más que todo funcionara.

- Luego, por separado, programé y diseñé gráficamente la pantalla de juego. En ella puse el tablero y toda la información necesaria para la partida. Luego agregé los gusanos (sin animar) y empecé con la programación del juego de dos personajes. Tras probar que todo funcionaba bien puse el código junto al anterior. Posteriormente agregé las animaciones de los gusanos y que cambiaran de color y complemento según lo escogido. Añadí que se pudiera poner el nombre de jugador que quisieramos y luego comprobé la correción de lo añadido.

- Lo siguiente fué que se pudiera elegir entre los 5 tipos de juegos diferentes. Una vez programado y siendo todo correcto, me puse a programar los 12 mapas por separado. Una

vez obtenido esto, lo añadí al código para comprobar que funcionaba correctamente.

- Luego programé las diferentes magias que se pueden usar en el juego y las 5 frecuencias en las que estas salen. Ya solo quedaba programar el boton de aleatorio y que se guardara en un fichero la configuración de la partida que habíamos seleccionado. Verifiqué que todo funcionaba correctamente.

- Lo siguiente fué programar la música y las opciones del juego. Una vez finalizado esto, llegamos a la parte que más trabajo me llevó: la inteligencia artificial. Que realizara exactamente lo que yo quería que hiciera fué realmente dificil, pero tras un trabajo duro conseguí que funcionara correctamente.

- Una vez finalizado esto, me puse con el bloqueo de las características del juego, que se van desbloqueando cuando vamos completando el juego contra la CPU. Tras funcionar correctamente empecé con los trofeos. Esta parte fué realmente sencilla y fué funcionando correctamente desde el principio.

- Una última cosa para terminar fué crear el tutorial del juego. Casi todo era diseño gráfico, asi que verificar que todo funcionaba era sencillo.

Una vez todo estaba programado, diseñado gráficamente y comprobada su funcionalidad, lo siguiente sería pasar a revisar que todo funcionaba correctamente. Para ello pasaríamos a las pruebas de errores, en la que no sólo comprobaría de nuevo que todo funcionaba como yo quería, sino que también comprobaría que pasaría si inserto errores a proposito.

7.2. Pruebas de errores

Una vez obtenido el sistema completo, me dediqué a realizar pruebas de errores o testeo del software. Para esta parte probé distinto tipos de usuarios:

-La primera prueba la realizé yo. Intenté buscar errores en el juego.

-Como yo conocía el código me era más dificil detectar errores, por lo que opte a que un compañero (informatico) buscara errores por mí.

-Una última prueba fué dejar el juego en manos de usuarios sin conocimientos de informática.

Estas distintas pruebas me sirvieron para encontrar algunos errores de codificación que no había encontrado antes. También me sirvieron para dar un enfoque diferente a algunos componente del juego (como unos menus mas interactivos y animados).

Capitulo 8

Conclusiones

Uno de los objetivos principales que me impuse al empezar este proyecto era saber desenvolverme en el desarrollo de los videojuegos. Uno de mi mayores logros ha sido aprender todo lo necesario para ello. Para saber más del juego se puede acudir a la página http://aaww.forja.rediris.es/.

8.1. Desarrollo del proyecto

Durante el proyecto he utilizado difentes tecnologias:

- C++: Este lenguaje lo aprendí durante la carrera, y es uno de los lenguajes más importantes para el desarrollo de los videojuegos. Evidentemente me ha resultado muy cómodo utilizar este lenguaje para mi proyecto.

- Gosu: Los conocimientos básicos de Gosu los adquirí durante de asignatura Diseño de Videojuegos, pero aún me quedaba profundizar más en sus conocimientos. La documentación existente es un poco escueta y con faltas de ejemplos, pero aún así es muy fácil de usar, ya que todo está totalmente orienta a desarrollos de videojuegos. Es una herramienta muy útil y sencilla se usar.

- GIMP: No es el programa más fácil de usar para edición de imágenes, pero es el mejor que puedes encontrar de licencia libre.

- Planner: Es muy útil, pero una herramienta demasiado potente para la envergadura de este proyecto. Opino que este proyecto no le puede sacar todo el jugo que se podría a esta herramienta.

- Tiled Map: Este editor de mapas me ha servido durante todo el proceso de edición de imágenes. Con el se puede trabajar las imágenes por capas. Esto facilita a los que no somos profesionales del dibujo a poder trabajar con las imágenes por componentes pequeños y a reutilizar las imágenes.

- Graphics Gale: Este programa de edición de pixeles es bastante sencillo de usar. Es simple, útil y no da ningún problema. Totalmente recomendado a aquellos que quieran trabajar editando pixeles.

Al terminar el juego quedé totalmente satisfecho con él. Al principio la idea que concebía de este juego era mucho mas simple y con menos cosas, pero a medida que fuí desarrollandolo me permití hacerlo más complejo y muchísimo más diverso. Esto hará que tengamos que estar jugando más horas para dominar totalmente el juego.

8.2. Posibles ampliaciones y mejoras

Aunque el juego está acabado, siempre se pueden mejorar cosas y ampliarlo. Haremos una lista de las posibles mejoras y ampliaciones que considero que se pueden hacer sobre el proyecto.

- Ampliaciones:
  - Nuevas fases: Como a todos los juegos, se puede ampliar introduciendo nuevas fases que resulten nuevos retos para los jugadores.
  - Nuevas magias: Incluir nuevas magias en el juego haría que este cambiara totalmente, ya que cambiarías las posibilidades y las tácticas a seguir en el juego.
  - Más trofeos: Esto serían más retos para el jugador, más horas de juego y mucha más diversión.

- Mejoras:
  - Editor de mapas: Nuevos mapas siempre es bueno, pero poder editar tus propios mapas sería sin duda algo definitivo para enganchar al juego a muchos jugadores.
  - Editor de gusanos: Crear un editor de gusano, en el que se pueda personalizar el gusano con el color deseado, su complemento, elegir rasgos faciales, pelos, incluso estatura y complexión sería lo ideal.

Cualquier otra mejora o ampliación sugerida por los jugadores sería una de las mejores que podriamos hacer, ya que al fin y al cabo son ellos los que disfrutan de este juego.

8.3. Objetivos

Al finalizar el proyecto siempre es conveniente hacer un repaso de aquellos objetivos que queríamos alcanzar, para comprobar si hemos alcanzados dichos objetivos:

- **Aprender a utilizar Gosu:** Me ha permitido saber los puntos fuertos de Gosu y todo aquello que se podría mejorar para facilitar la utilización de este. Me ha resultado dificil a veces entender cómo usar algunas de las clases de Gosu debido a la falta de ejemplos en la documentación de este.

- **Aprender todo sobre el desarrollo de un videojuego 2D:** Al haber realizado el análisis del sitema, el diseño, la implementación y la elaboración gráfica he podido aprender todo lo necesario para desarrollar un videojuego en 2D. Aunque pienso que siempre quedan cosas por aprender.

- Comprobar que puedo desenvolverme sólo ante nuevas lirerías/lenguajes. Las primeras librerías que usas puede resultar más complejas y llevarte más tiempo, pero luego aprendes cómo profundizar en ellas.

- Poder solventar todos los problemas que me surjiesen durante el desarrollo es uno de los objetivos más importante para mí, ya que de estos problemas son de los que más se aprenden.

Apéndice A

Guía de Instalación (Linux)

Antes de descargar y compilar juego debemos preparar el equipo para ello. Primero instalaremos gcc y make para poder compilar el codigo:

sudo apt-get install gcc g++ make

Ahora instalaremos el Subversión para poder descargar el juego:

sudo apt-get install subversion

Ahora instalaremos distintas librerias para el manejo de gosu:

sudo apt-get install build-essential
sudo apt-get install freeglut3-dev
sudo apt-get install libfreeimage-dev
sudo apt-get install libgl1-mesa-dev
sudo apt-get install libopenal-dev
sudo apt-get install libpango1.0-dev
sudo apt-get install libsdl-mixer1.2-dev
sudo apt-get install libsdl-ttf2.0-dev libsndfile-dev
sudo apt-get install libxinerama-dev
sudo apt-get install libsndfile-dev

Ahora descargamos el juego:

sudo checkout https://forja.rediris.es/svn/aaww

Tras haberlo descargado entero tendremos que descomprimir y configurar Gosu:

cd aaww/documentacion/
tar xvzf gosu.tar.gz
cd ..
make libgosu

Ahora compilamos el juego:

make

Por último sólo tenemos que ejecutar el juego:

./Apples_wars

Espero que disfrutes del juego.

Apéndice B

Guía del Usuario

Figura B.1: Logo

Introducción

Apples Wars es un videojuego de estrategia-tablero. El juego consiste en obtener puntos (dependiendo de la modalidad) manejando un grupo de carismáticos gusanos. Estos puntos se obtendrá robando manzanas a los demás gusanos (ya sean tuyos o del oponente).

Podrás caracterizar tus gusanos como más te guste, jugar en distintos escenarios (con un amigo o contra la CPU) e includo usar magias para fortificar tus jugadas.

Deberás establecer distintas estrategias adaptandote a los diferentes retos que suponen los distintos modos de juego y los distintos escenarios.

Reta a tus amigos a una partida y descubre quien es el mejor estratega.

Gracias por adquirir Apples Wars.

Controles

Lo único que necesitaremos para jugar a este juego será un ordenador con ratón.
En cualquier momento se podrá salir del juego pulsando la tecla ESC.

Tutorial

Una vez entrado en el juego, en el menú principal podrás escoger la opcion TUTORIAL, en la que te explicará detalladamente cómo jugar y manejar cada rincón de este juego.

Figura B.2: Tutorial

A continuación se incluye las imagenes del tutorial:

Figura B.3: Tutorial 1

Menús
Pulsar aquí para jugar contra un amigo (2 jugadores)
PLAYER VS PLAYER
PLAYER VS CPU
Pulsar aquí para volver al menú anterior
Pulsar aquí para jugar contra la computadora (1 jugador)
Volver
Figura B.4: Tutorial 2

Figura B.5: Tutorial 3

Figura B.6: Tutorial 4

Nombre: Munt Gagu
Color:
Acc.:
Pulsar aquí para cambiar el nombre del jugador
Pulsar sobre el color deseado para nuestro gusano (no puede coincidir con el color del otro jugador)
Pulsar sobre el complemento deseado para nuestro gusano (para deseleccionar el complemento volver a pulsar sobre el mismo complemento)
Muestra como será nuestro gusano
Selección
Figura B.7: Tutorial 5

Selección
Nombre: CPU
Color:
Nivel:
Fácil
Normal
Para la CPU no se puede elegir nombre
Pulsar sobre el color deseado para el gusano de la CPU (no puede coincidir con el color del otro jugador)
Pulsar sobre la dificultad:
- Fácil: La CPU realiza movimientos básicos.
- Normal: La CPU realiza mejores movimeintos y las mágias instantáneas.
- Dificil: La CPU realiza movimientos complejos y todo tipo de mágias (instantáneas y de selección).
Muestra como será el gusano de la CPU
Figura B.8: Tutorial 6

JUEGO
Puntos
Crisálidas
Niveles
Gusanos
Supervivencia
Acumula puntos al completar una crisálida
El total de puntos es la suma de los gusanos que queden
Tipo de juego
Acumula puntos al robar a tu enemigo su manzana
El total de puntos es la suma de las manzanas de tus gusanos
Acumula puntos al robar a tus gusanos su manzana

Figura B.9: Tutorial 7

MAGIAS
Crisálida
5 turnos
3 turnos
1 turno
Supervivencia
Consigues una mágia cada 5 turnos
Consigues una mágina cada turno
Frecuencias de mágias
Consigues una mágia al hacer una crisálida
Consigues una mágia cada 3 turnos
Se consigue una mágia al hacer una SUPERVIVENCIA (coger manzanas de tu equipo)

Figura B.10: Tutorial 8

Figura B.11: Tutorial 9

Figura B.12: Tutorial 10

Mágias
Hay 2 tipos de mágias:
- De selección: son las que se selecciona un gusano.
- Instantáneas: son las que se activan sin seleccionar ningún gusano.
Nombre de la mágia
Icono de la mágia
Nacimiento
Puedes poner una crisálida de tu equipo
Descripción de la mágia
ACTIVAR
Aquí se pulsa para activar y desactivar la mágia
Figura B.13: Tutorial 11

Turno de juego
El juego se realizará por turnos. En cada turno moverá un jugador diferente. El turno tiene dos fases:
1- Uso de mágia. No es obligatorio. Cuidado porque sólo podemos tener una mágia a la vez.
2- Movimiento. Seleccionamos un gusano (ya sea tuyo o del enemigo) y realizamos un movimiento. El gusano seleccionado tiene el área verde y el movimiento posible tiene área azul.
Turno
Munt Gagu
Puntuación
Magia
Sin magia
No tienes ninguna magia actualmente
Salir
Figura B.14: Tutorial 12

Conseguiremos puntos al realizar el movimiento correcto respecto al tipo de juego elegido
Movimientos
Turno
Magia
Robar
Hay 3 tipos de movimeintos:
- Normal: Cuando mueves un gusano de tu equipo al gusano del otro equipo.
- Crisálida: Al conseguir 5 manzanas, y al realizarla obtenemos puntos extras.
- Supervivencia: Cuando mueves un gusano de tu equipo a otro de tu equipo.
Salir
Figura B.15: Tutorial 13

Figura B.16: Tutorial 14

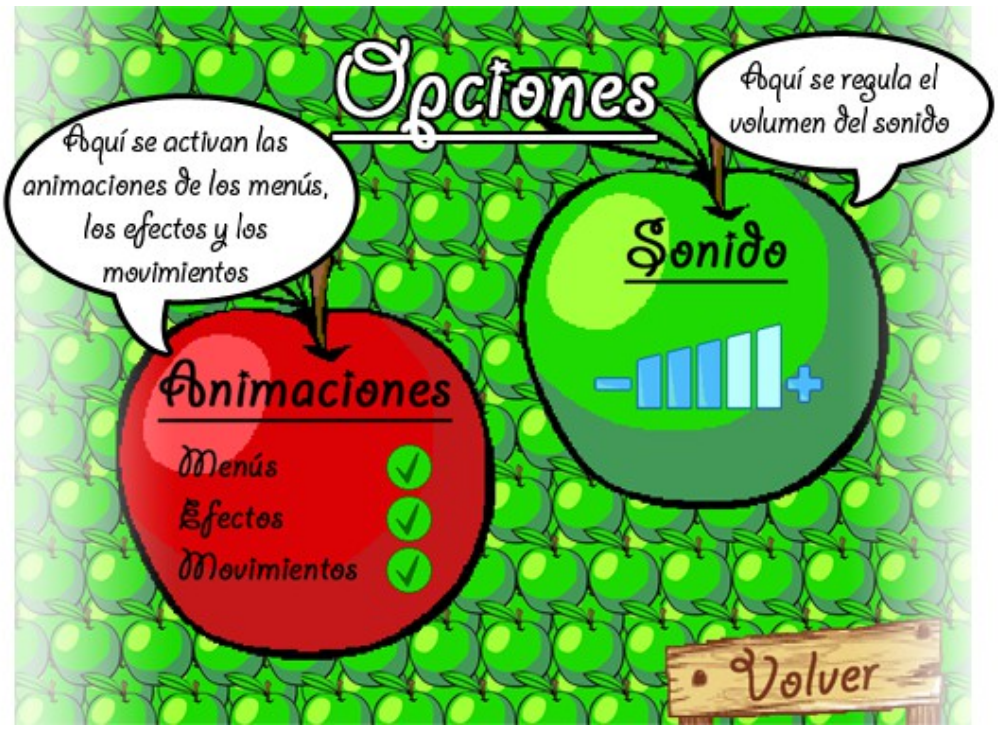

Figura B.17: Tutorial 15

Apéndice C

Tecnologías implicadas

En esta sección comentaré las herramientas que he utilizado para el desarrollo del proyecto:

- C++: Lenguaje de programación orientado a objetos. Es el lenguaje principal del juego.
- Gosu: biblioteca de desarrollo de videojuegos 2D multiplataforma, software libre. [2]
- GIMP: Programa de edición de imágenes digitales en forma de mapa de bits, tanto dibujos como fotografías. Se ha usado para el retoque de algunas imágenes y la realización de otras.
- Planner: Herramienta libre multiplataforma para la gestión y planificación de proyectos. De aquí he obtenido el diagrama de Gantt.
- Tiled Map: Editor de mapas. Lo he usado para diseñar los mapas a través de las plantillas creadas. [3]
- Graphics Gale: Programa de edición de pixeles. Lo he usado para crear las animaciones del juego y las plantillas que darían forma a los mapas del juego. [4]

Apéndice D

Reglas del juego

En esta sección comentaré las reglas para jugar al juego:

- Podremos elegir entre 5 tipos de juegos. Lo que diferenciará a estos tipos de juegos es la forma de conseguir puntos:
  - Puntos: Conseguiremos un punto al quitarle a un gusano del oponente una manzana.
  - Crisálidas: Conseguiremos un punto al hacer una crisálida.
  - Niveles: Conseguiremos un punto por cada manzana que tenga los gusanos de nuestro equipo.
  - Gusanos: Conseguiremos un punto por cada gusano.
  - Supervivencia: Conseguiremos un punto al quitarle una manzana a un gusano de nuestro equipo.
- Habrá que jugar hasta que no queden más movimientos para realizar.
- Las magias siempre hay que usarlas antes de mover.
- Las magias serán asignadas al azar según la frecuencia de magias escogida:
  - Crisálida: Conseguimos una magia al realizar una crisálida.
  - 5 turnos: Cada 5 turnos se le asigna una magia a un jugador diferente.
  - 3 turnos: Cada 3 turnos se le asigna una magia a un jugador diferente.
  - 1 turno: Cada turno se le asigna una magia a un jugador diferente.
  - Supervivencia: Conseguiremos una magia al realizar una supervivencia.
- Sólo podremos tener 1 magia. Al conseguir otra perderemos la anterior.
- Un gusano sólo podrá acumular un máximo de 5 manzanas, convirtiendose en crisálida al conseguirlas.
- Podremos mover cualquier gusano, aunque sea de nuestro oponente.
- Podremos mover en cualquier dirección, siempre que en dicha dirección haya un gusano y la suma de sus manzanas no supere el máximo permitido para un gusano.
- No podemos mover un gusano hacia una crisálida ni a un posición vacia.
- En un turno sólo podremos realizar una magia (si tenemos) y un movimiento.
- Los dos jugadores han de tener nombre distintos.
- No podrán escoger el mismo color de gusano ambos jugadores.
- El jugador que comienza a jugar será determinado al azar.
- El tablero será de dimensiones 8x8.
- Las fichas se colocarán entremezcladas, de forma que un gusano del jugador 1 siempre tenga encima, debajo, a la derecha y a la izquierda un gusano del oponente.
- Ganará aquel jugador que más puntos tenga al finalizar la partida.

Bibliografías y referencias

[1] Varios autores. www.musicalibre.es

[2] www.libgosu.org

[3] www.mapeditor.org

[4] www.humanbalance.net/gale/us

[5] Varios autores. http://es.wikipedia.org/wiki/Juego_de_estrategia

[6] Varios autores. http://es.wikipedia.org/wiki/Videojuego#Principales_g.C3.A9neros

[7] CCOO (2010). Tablas salariales 2010 IV Convenio Colectivo. http://www.uca.es/sindicato/ccoo/documentos/tabla-salarial-pas-laboral-2010.pdf

[8] Descargar el juego. https://forja.rediris.es/scm/?group_id=1109

[9] Francisco Palomo Gerardo Aburruzaga, Inmaculada medina. Fundamentos de C++. Servicio de Publicaciones de la Universidad de Cádiz, 2009.

[10] Jim Thompson, Bernaby Berbank-Green, Nic Cusworth. Videojuegos : manual para diseñadores gráficos. Gustavo Gili, 2008.

[11] Roger S. Pressman ; traducción Jesús Elmer Murrieta Murrieta, Eloy Pineda Rojas, Víctor Campos Olguín ; revisión técnica Raúl Antonio Trejo Ramírez, Saturnina Teodora Juárez Ramírez. Ingeniería del software : un enfoque práctico. McGraw-Hill, 2010.

[12] Estudio de los videojuegos como medio didácticos. http://www.uib.es/depart/gte/calvo.html

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software

inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the
public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free
programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require,
such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your
rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works
for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with

you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code;
keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified
it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section
7. This requirement modifies the requirement in section 4 to "keep intact all notices".

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices,

your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not
used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or

sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the
Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately
under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains
a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means
prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do
not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever
licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that
any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not

include claims that would be infringed only as a
consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the
patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that
country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered
work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is
in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work
conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

<program> Copyright (C) <year> <name of author>

This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary.
For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.